UNIVERSITÉ DE MONTRÉAL

A MULTIRESOLUTION SUBDIVISION-BASED FRAMEWORK FOR INTERACTIVE SURFACE MESH EDITING

MAN WANG
DÉPARTEMENT DE GÉNIE INFORMATIQUE
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

MÉMOIRE PRÉSENTÉ EN VUE DE L'OBTENTION
DU DIPLÔME DE MAÎTRISE ÈS SCIENCES APPLIQUÉES
(GÉNIE INFORMATIQUE)
AOÛT 2006

Your file Votre référence
ISBN: 978-0-494-19341-9
Our file Notre référence
ISBN: 978-0-494-19341-9

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Ce mémoire intitulé:

A MULTIRESOLUTION SUBDIVISION-BASED FRAMEWORK FOR INTERACTIVE SURFACE MESH EDITING

présenté par: WANG Man

en vue de l'obtention du diplôme de: Maîtrise ès science appliquées

a été dûment accepté par le jury d'examen constitué de:

M. GRANGER LOUIS, M.Sc., président

M. GUIBAULT François, Ph.D., membre et directeur de recherche

M. KHACHAN Mohammed, Ph.D., membre

For all who once help me;

especially for my husband David

ACKNOWLEDGEMENT

Writing this thesis is a kind of epic journey for me. However, it is fortunate that I haven't had to adventure it alone. First and foremost, I would like to express my sincere appreciation to my director, Professor François Guibault, for his guidance in pursuing this research and his instructing in writing this thesis. The constant and prominent amelioration of this thesis is completely attributed to my director's intuitive advices and arduous corrections.

Special thanks are also due to Professor Louis Granger and Dr. Mohammed Khachan for their patience of reading and evaluating this thesis, Professor John Mullins and Professor Alejandro Quintero for their encouragement.

Besides, my thanks must go to Qinghu Liao and Jean-François Dubé who shared experience in relevant research and recommended useful references. As for extolling the virtues of making my graduate student experience at École Polytechnique more pleasurable, I'd like to extend my thanks to Mélissa Côté and Yun Wang.

It is my husband David whose passion for his job always inspires me to work harder. Particularly, it is him who makes my life more colourful and more hopeful. Even though once enduring difficulties in my study, my mind seems wholly taken up with reminiscences of past gaiety, because I experienced them with him together. I really want to thank him for his incredible

contribution in thousands of ways, especially his love and his support.

Last but not least, I would like to thank again all the members in my family. Most of all, my parents, whose immense love support me everywhere; my parents-in-law and my sister-in-law Isabelle, who always bring me so much courage and care that I am not so nostalgic.

All of them have my deepest gratitude for their unstinting support.

RÉSUMÉ

Ce mémoire présente les résultats d'une recherche visant à concevoir une représentation et des outils pouvant former un environnement d'édition de surfaces de subdivision hiérarchiques incorporant une certaine interactivité et flexibilité dans le contrôle et la modification des détails visuels. Ce travail se concentre sur deux schémas de subdivision: *Loop* et *Modified Butterfly*. Une comparaison entre les surfaces approximées avec le schéma *Loop* et les surfaces interpolées avec le schéma *Modified Butterfly* est effectuée. Notre recherche se concentre sur l'amélioration des algorithmes et des structures de données permettant la manipulation des surfaces de subdivision sur différents niveaux de manière efficace et interactive. Notre but est de construire un système d'édition où les utilisateurs peuvent manipuler des surfaces de façon rapide et intuitive.

Nous nous penchons sur quatre aspects de base pour la réalisation d'un tel environnement. (1), L'exécution des techniques de subdivision, lesquelles génèrent des surfaces lisses issues d'une séquence de maillages polyédraux raffinés, est une base importante de cette structure de subdivision. (2), La comparaison entre les maillages de surface générés à partir des schémas de *Loop* et de *Modified Butterfly* est analysée. (3), L'efficacité à modifier et à afficher des surfaces de subdivision représentant une géométrie complexe nous amène à choisir et à améliorer certains algorithmes d'édition. (4), L'édition interactive nous pose aussi un problème sur la manipulation des

arêtes vives, lesquelles définissent les caractéristiques des maillages multi-résolution des surfaces de subdivision. La définition de plis par l'usager est un point crucial dans un environnent d'édition de surfaces de subdivision.

Notre recherche débute en considérant un éventail de modèles d'objets complexes représentés en 3D. Après avoir rendu et affiché un modèle, nous pouvons subdiviser ce maillage de base en un maillage plus raffiné. Nous pouvons également modifier le maillage à différents niveaux en déplaçant les points de contrôle. Aussi, il nous est possible de définir des arêtes vives afin de rencontrer certaines exigences spécifiques. Les avantages de ce cadre d'édition de maillage de surface seront démontrés en comparant les modifications de plusieurs modèles 3D et les coûts de calcul pour chacun des schémas.

Le langage de programmation C++ ainsi que la librairie graphique OpenGL forment la base de l'environnement de notre projet. MATLAB est également utilisé pour l'analyse de résultats. De plus, Rhinoceros est utilisé pour nous permettre d'étudier la modélisation et l'analyse de la précision et de la compatibilité des maillages de surface.

ABSTRACT

This thesis presents a multi-resolution framework for editing hierarchical subdivision-based surfaces that achieves a certain level of interactivity, modification capacity and control over visual detail. This research mainly considers two subdivision schemes: Loop subdivision and Modified Butterfly subdivision. The comparison between approximating surface meshes generated by Loop subdivision and interpolating surface meshes generated by Modified Butterfly subdivision are elaborated upon. Our research focuses on improving algorithms and data structures which permit to manipulate subdivision-based surface meshes at multiple levels efficiently and interactively. Our goal is to build an editing system for users to effectively and intuitively interact with model surfaces.

Four basic contributions are addressed in this work: (1) Implementing subdivision technique, which successively generates smooth surfaces as a sequence of refined polyhedral meshes, is an important technical foundation of our multi-resolution framework; (2) Comparing surface meshes respectively generated by Loop subdivision and by Modified Butterfly subdivision is detailed; (3) The efficiencies of modifying and rendering complex subdivision surfaces challenge us to choose and ameliorate appropriate editing algorithms; (4) Interactive editing also brings forth the handling of creases, which adjust sharp features on multi-resolution subdivision-based surfaces along with a set of user-defined edges. The work

of creating creases presents a crucial application within the subdivision framework.

Our research starts by considering various complex 3D object representations. It then turns to subdivision approaches and considers hierarchical representation approaches. Next, multi-level modifications are introduced and relevant data structures are presented. Finally, creases are designed in order to meet special modeling requirements. The benefits of this mesh-based framework will be demonstrated by comparing various modified geometrical models and time costs used to perform each scheme.

The C++ programming language and OpenGL supplies the basic environment for developing our project. Meanwhile, MATLAB is used for analyzing the results of system implementation. In addition, Rhinoceros is a useful tool to help us in modeling and analyzing accuracy and compatibility of mesh models.

CONDENSÉ

1. Introduction

Les maillages de surface sont largement utilisés pour la simulation, la visualisation et l'optimisation de la représentation d'objets physiques. Leur utilisation s'étend du design industriel et manufacturier jusqu'aux organismes physiologiques pour la planification de chirurgie, en passant par les modèles mentaux d'interprétation d'images ultrasoniques ainsi que les phénomènes physiques tel que les effets et les forces exercés par des variations électriques et magnétiques. Ces utilisations constituent une partie importante des outils de modélisation 3D. Cependant, malgré tous les progrès réalisés dans ce domaine, la gestion des maillages de plusieurs milliers de triangles dans un environnement interactif pose toujours un problème de taille pour les systèmes de modélisation et d'affichage. Ces problèmes de performance ont toujours été une préoccupation depuis l'existence des systèmes de modélisation. En plus de chercher à maintenir un équilibre entre la qualité et la vitesse, les recherches visant à améliorer les méthodes d'édition des maillages de surface de manière efficace jouent un rôle significatif dans le processus de modélisation géométrique.

Notre projet vise à approfondir les approches d'édition de maillages de surface multi-résolution à travers le développement d'un environnement

intégrant différents schémas de subdivision. L'approche de modélisation utilisée devrait permettre aux utilisateurs d'interagir avec des surfaces de subdivision sur plusieurs niveaux de détail.

Basée sur la subdivision, notre recherche contient trois parties interdépendantes : les schémas de subdivision, la technologie de multi-résolution et les fonctions d'édition interactive. Toutes ces parties sont basées sur des modèles de maillage de surface 3D et l'aspect d'édition interactif des surfaces sera approfondi.

Notre mémoire est constituée de quatre parties principales. « Revue bibliographique » fournit les références pour les concepts de base des représentations de surface, des surfaces de subdivision et des technologies de multi-résolution et LOD. Dans ce chapitre, les représentations de surface et les schémas de subdivision sont présentés et classifiés. De plus, un aperçu des applications utilisant les surfaces de subdivision, les technologies de multi-résolution et les LOD, techniques vitales en édition interactive, est donné.

Afin de couvrir la théorie et l'application, le chapitre intitulé « Exécution de l'édition de maillage de surface de multi-résolution » décrit les techniques utilisées pour générer des maillages de surface multi-résolutions et pour manipuler des maillages à différents niveaux de subdivision. Le but de ce chapitre est d'exécuter le schéma de *Loop* et le schéma de *Modified Butterfly* et de fournir une comparaison détaillée entre les schémas d'approximation et ceux d'interpolation. Nous sélectionnons de manière concrète plusieurs

méthodes d'édition de maillage de surface pour présenter notre cadre de subdivision et multi-résolution. À la fin de ce chapitre, nous présentons les structures de données et les algorithmes permettant l'édition locale tout en gardant les changements des divers niveaux.

Dans le chapitre « Évaluation et résultats », la performance de notre projet, basée sur des statistiques de coûts CPU découlant de manipulations spécifiques en temps réel, est évaluée. Les résultats pertinents sont analysés en regard des objectifs de notre recherche.

Le chapitre « Conclusion et recherches futures » contient un résumé de nos travaux et plusieurs points à approfondir sur les surfaces de subdivision, l'évaluation de la qualité des surfaces ainsi que l'amélioration des manipulations des maillages de surface.

2. Revue bibliographique

Nous situons notre recherche dans un vaste contexte d'ouvrages reliés à la représentation de surface, aux surfaces de subdivision, aux techniques multi-résolutions avec LOD (niveau de détails) et à l'édition interactive de maillage.

2.1 Représentations de surfaces

Les représentations de surfaces sont une partie essentielle des systèmes de

modélisation géométrique. Il y a quatre types communs de surface : les surfaces implicites, les surfaces paramétriques, les maillages polygonaux et les surfaces de subdivision. Chaque type comporte des points forts et des points faibles. Certains semblent mieux adaptés à quelques applications alors que d'autres réagissent mieux dans d'autres contextes.

Les surfaces implicites sont particulièrement bien adaptées afin de décrire des molécules, de calculer des intersections de surfaces pour les détections de collision et pour savoir si un point est à l'intérieur ou à l'extérieur d'une surface. Cependant, l'utilisation des surfaces implicites est plus compliquée et non intuitive pour représenter des surfaces qui doivent être modifiées interactivement.

Bien que la description des maillages polygonaux ne se limite à aucune topologie, elle devient rapidement trop lourde lorsqu'il y a abondance de détails. Lorsque nous interagissons avec des maillages polygonaux de grande complexité, le matériel informatique courant aura de la difficulté à accomplir les tâches demandées. Contrairement aux maillages polygonaux, les surfaces polynomiales peuvent être utilisées sans surcharger le matériel informatique. Toutefois, lorsque les surfaces polynomiales sont utilisées pour des détails plus fins, le coût de calcul des paramètres de contrôle devient très élevé.

2.2 Surfaces de Subdivision

Les surfaces de subdivision se retrouvent quelque part entre les topologies arbitraires des maillages polygonaux et les résolutions variables des surfaces

polynomiales. Elles sont excellentes pour générer une surface complexe et lisse issue d'une séquence de raffinement de maillage (Zorin et al. 2000).

Les caractéristiques des quatre types de surfaces communes sont indiquées et comparées à l'aide du tableau 1 (Pellacini 2005).

Tableau 1: Comparaison générale des représentations de surfaces (Pellacini 2005).

	Surface implicite	Surface paramétrique	Maillage polygonal	Surface de subdivision
Précision	Oui	Oui	Non	Oui
Concision	Oui	Oui	Non	Oui
Spécification intuitive	Non	Oui	Non	Non
Localité	Non	Oui	Oui	Oui
Invariance affine	Oui	Oui	Oui	Oui
Topologie arbitraire	Non	Non	Oui	Oui
Continuité garantie	Oui	Oui	Non	Oui
Paramétrisation	Non	Oui	Non	Non
Affichage rapide	Non	Oui	Oui	Oui
Intersection rapide	Oui	Non	Non	Non

Suite à la lecture du tableau ci-dessus, nous pouvons conclure que les surfaces de subdivision sont issues de primitives de surfaces d'ordre

supérieur. En nous référant au tableau 1, nous remarquons que les surfaces de subdivision peuvent définir de vastes sections en utilisant seulement quelques points de contrôle tout en permettant de préserver la continuité, contrairement aux maillages polygonaux. De plus, les surfaces de subdivision constituent une façon attrayante de modéliser des topologies arbitraires contrairement aux surfaces paramétriques. Donc, d'une certaine manière, les surfaces de subdivision peuvent être vues comme le lien et l'unification des surfaces paramétriques et des maillages polygonaux.

Toujours selon le tableau 1, il semble que les surfaces de subdivision soient désavantagées pour la paramétrisation et la spécification intuitive par rapport aux surfaces paramétriques. Cependant, à strictement parler, la paramétrisation ne fait pas partie des surfaces de subdivision. Si nous tentons d'améliorer les capacités d'édition interactive des surfaces de subdivision, la spécification intuitive doit s'améliorer. Afin de concrétiser cette amélioration, nous nous concentrons sur les capacités d'édition interactive des surfaces de subdivision multi-résolutions.

En outre, les caractéristiques des surfaces de subdivision touchent plusieurs importants problèmes informatiques et topologiques auxquels sont confrontés les spécialistes en graphisme. La motivation d'utiliser les surfaces de subdivision provient également de leurs caractéristiques, lesquelles peuvent être résumées ainsi : simplicité, efficacité, topologie arbitraire, uniformité des représentations, scalabilité et stabilité numérique.

2.3 Travaux connexes sur les techniques de multi-résolution et de LOD

Les maillages de surface multi-résolutions fournissent un cadre structural pour générer, manipuler et effectuer le rendu des représentations de modèles géométriques complexes à divers niveaux de détails. Les technologies de multi-résolution incluant plusieurs niveaux de détails sont utiles dans des applications graphiques en temps réel pour, par exemple, améliorer la visualisation de simulations médicales ainsi que pour accélérer le rendu de système d'édition interactif et augmenter l'authenticité de système de réalité virtuelle pour l'éducation ou le divertissement.

Les travaux liés aux techniques de multi-résolution peuvent être classifiés en deux catégories : simplification de surface et amélioration de surface. La simplification de surface se résume à prendre un maillage détaillé comme maillage original et à en extraire plusieurs maillages à différents niveaux en utilisant des algorithmes de simplification. L'amélioration de surface consiste quant à elle à prendre un maillage simple comme maillage d'origine puis à le raffiner vers une version plus dense en appliquant des algorithmes de subdivision.

Dans Zorin et al. (2000) et Warren et Weimer (2002), les différentes approches de multi-résolution par subdivision sont décrites en détail. Un bon exemple d'exécution pratique de ces techniques de subdivision peut être trouvé dans Brickhill (2001).

Les techniques de "Level of Detail" (LOD) sont reliées à la précision des

maillages approximant un modèle géométrique (Armin et al. 2002, Floriani et Magillo 2002). L'utilisation la plus commune de LOD est le rendu de versions simplifiées d'objets distants permettant de réduire le nombre de détails (Luebke et al. 2003). L'histoire des techniques de LOD suit de près le développement des techniques multi-résolutions.

2.4 Travaux connexes sur l'édition interactive de maillage

L'utilisation de maillages polygonaux pour représenter des surfaces est omniprésente dans les outils de modélisation géométrique. Afin de pouvoir visualiser plusieurs niveaux de détails d'un maillage, les techniques de multi-résolution sont utilisées pour gérer les approximations sur plusieurs niveaux. En outre, en raison des conditions requises pour la manipulation de maillages de surface, les méthodes d'édition interactives de maillage ont besoin d'une grande qualification pour le remaillage et l'affichage de maillage de très grande taille, tout en restant flexibles et efficaces. C'est pourquoi les recherches sur les techniques multi-résolutions pour l'édition interactive de maillage sont au premier rang parmi les sujets de recherche dans les modeleurs géométriques 3D. Les questions portant sur l'édition interactive de maillage multi-résolution peuvent être décrites d'une part par les opérations d'édition et de déformation, et d'autre part par les structures de données et les algorithmes.

2.5 Réitération de notre recherche

Dans les sections précédentes, un survol de travaux connexes couvrant ce

vaste champ de recherche est présenté. Notre recherche ne s'inspire pas seulement de ces avancées techniques, elle les étend au-delà des travaux déjà présentés.

Dans notre projet, nous avons choisi deux schémas de subdivision comme base pour générer des maillages de surface à multi-résolutions : *Loop* et *Modified Butterfly*. Nous utiliserons le schéma de *Loop* comme représentation d'un schéma d'approximation et le schéma de *Modified Butterfly* comme représentation d'un schéma d'interpolation. Nous analyserons leurs différences en détails.

En nous basant sur d'anciens travaux à propos des frontières et plis, nous améliorerons les masques utilisés en vérifiant leur validité à l'aide de maillages de topologie variée. De plus, nous spécialiserons l'utilisation des plis de deux façons : plis d'arête et plis de sommet.

Nous utiliserons un *arbre quaternaire* similaire à celui décrit par Zorin et al. (2000) comme structure de base pour notre projet. Au niveau de l'exécution, l'innovation de nos travaux réside dans l'évaluation des performances des deux schémas de subdivision et dans le résumé des coûts CPU lors de l'exécution de tâches spécifiques.

Pour augmenter l'intuitivité des manipulations des surfaces de subdivision, l'algorithme d'édition de maillages est amélioré. En contraste avec les travaux antérieurs sur le sujet, la nouveauté de notre projet se situe au niveau de la sauvegarde des changements d'édition de tous les niveaux lors

des opérations de subdivision, non seulement à partir des niveaux inférieurs (héritage du niveau parent), mais aussi à partir des niveaux supérieurs (rétrospection des enfants), tout en conservant l'efficacité du support local utilisé dans notre cadre de multi-résolution et LOD.

Tout en favorisant une approche intuitive au niveau de l'interface graphique (GUI ou Graphical User Interface), notre projet explore des systèmes d'interface pour afficher et manipuler les maillages de surface et plusieurs fonctionnalités d'édition interactive. La contribution de nos travaux vise à étendre les différentes méthodes de manipulations interactives des maillages de surface.

3. Exécution d'édition de maillage de surface multi-résolution

Dans notre projet, nous avons codé deux schémas de subdivision de type « subdivision de facette » : le schéma de *Loop* et le schéma de *Modified Butterfly*. Une subdivision de type « subdivision de facette » peut-être décrite comme l'action consistant à subdiviser un maillage de triangles en divisant chaque triangle en quatre nouveaux triangles plus petits. Les deux schémas insèrent les nouveaux sommets à chaque arête des triangles et connectent ces nouveaux sommets pour un niveau de subdivision donné. Ces sommets additionnels sont appelés *sommets impairs* alors que les anciens sommets sont appelés *sommets pairs* (Zorin et al. 2000). Les deux schémas de subdivision peuvent être résumés comme consistant en deux procédés : (1) Subdiviser le maillage pour générer les nouveaux triangles et insérer les

nouveaux sommets, ce qui mettra également à jour la topologie du maillage; (2) Appliquer les règles de calcul correspondantes pour calculer la position des sommets, ce qui définira la géométrie du maillage.

Le schéma de *Loop* est un schéma d'approximation générant des surfaces avec une continuité C^2 aux endroits où la topologie du maillage initial est régulière. Cependant, la continuité sera C^1 là où se trouvent des sommets extraordinaires. Différemment, le schéma de *Modified Butterfly* est un schéma d'interpolation produisant des maillages de surface de topologie arbitraire avec une continuité C^1 dans la plupart des cas (Zorin 1998). Les différentes règles de subdivision utilisées pour calculer la position des sommets pour ces deux schémas produisent des résultats visuels significativement différents.

3.1 Schéma de Loop

Pour le schéma de subdivision de *Loop*, chaque sommet intérieur impair est influencé par les deux sommets aux extrémités de l'arête et par les deux autres sommets opposés à l'arête. Cela définit les deux triangles voisins se partageant cette arête. De plus, chaque sommet intérieur pair est influencé par sa propre position ainsi que celle de ses sommets adjacents. Quant aux sommets de bord, pairs et impairs, leurs masques ont été proposés par Hoppe et al. (1990).

Afin de permettre aux surfaces de contenir certaines arêtes non adoucies, des arêtes intérieures peuvent être étiquetées comme plis. Ces arêtes intérieures

désirées vives sont alors calculées comme des sommets de bord en réduisant l'influence de leurs sommets voisins durant l'étape de subdivision. Pour notre projet, nous définissons certains plis afin de manipuler et de modifier les maillages de surface. Finalement, selon le type d'influence des sommets voisins, nous définissons deux types de plis : plis d'arête et plis de sommet.

3.2 Schéma de Modified Butterfly

Le schéma de *Modified Butterfly* est un algorithme de type interpolation, ce qui implique qu'il ne change pas la position des sommets pairs. En contraste avec le schéma de *Loop*, les masques de *Modified Butterfly*, pour calculer les sommets impairs, sont beaucoup plus compliqués. On en distingue trois types pour les sommets impairs : *(a)* avec voisins réguliers, *(b)* avec voisins réguliers de bord ou de pli, et *(c)* avec voisins d'intérieur non réguliers, de bord ou de pli (Zorin et al. 1998). En plus, le type *(b)* inclut au moins quatre sous-types de masque pour certains cas spéciaux. Et le type *(c)* inclut deux sous-types différents déterminés par le type de sommet voisin, ce qui a un impact sur le résultat du schéma dans son ensemble.

Pour notre recherche, nous exécutons le schéma de *Modified Butterfly* selon Zorin et al. (2000) en premier. Cependant, les résultats visuels n'étaient pas idéaux. Durant l'évolution de notre projet, nous avons trouvé que certains coefficients dans les masques de Zorin et al. (2000) n'avaient pas été correctement spécifiés. Par la suite, nous avons trouvé qu'une version améliorée avait été proposée par Attene et al. (2005), qui présentaient une façon de récupérer les éléments vifs et pointus des maillages. Finalement,

nous avons amélioré notre version grâce à des changements provenant des deux références et nous les avons validés en utilisant plusieurs modèles variés, en particulier ceux composés de bordures. Les résultats finaux se sont avérés efficaces pour l'atteinte d'un lissage correct sur les maillages de surface.

3.3 Comparaison entre les deux schémas de subdivision

Après recherches et analyses, nous arrivons à la conclusion que même si le schéma de *Modified Butterfly* prend plus de temps de calcul pour les masques, le coût demeure négligeable en comparaison avec le temps de calcul nécessaire pour la gestion de la mémoire, le remplissage des structures et la connectivité.

Dans le schéma de *Loop*, les sommets pairs ne sont pas fixes. Il en découle que les surfaces de subdivision de niveau plus élevé ne passent pas par les points de contrôle générés aux niveaux inférieurs. Les plis sont donc plus apparents.

Comparé au schéma de subdivision de *Loop*, le lissage du schéma de *Modified Butterfly* n'est pas aussi satisfaisant. Il est donc plus difficile de valider le lissage et l'exécution du schéma, surtout lors de l'utilisation des masques pour sommets irréguliers, lesquels sont très diversifiés.

La surface ultime du schéma de *Loop* ressemble à un rétrécissement du maillage original qui converge vers une surface entièrement contenue dans

les limites du maillage original, alors que la surface ultime du schéma de *Modified Butterfly* ressemble à une amplification du maillage original. C'est pourquoi la rapidité de convergence du schéma de *Loop* est plus élevée et les formes du maillage final sont plus petites que les originaux, alors qu'avec le schéma de *Modified Butterfly*, elles sont légèrement plus grosses.

Lorsque les maillages de surface originaux sont plus denses et possèdent moins de bords, l'utilisation du schéma de *Modified Butterfly* peut atteindre un fini de lissage supérieur. Donc, afin d'obtenir un bon lissage avec le schéma de *Modified Butterfly*, la densité et la topologie du maillage original sont primordiales. Nous avons trouvé qu'une excellente combinaison consiste tout d'abord à appliquer le schéma de *Loop* sur le maillage original afin d'obtenir une plus grande densité, et ensuite à appliquer le schéma de *Modified Butterfly*. Les deux schémas de subdivision ne sont donc pas incompatibles et au contraire peuvent être utilisés ensemble afin de générer de meilleurs résultats.

3.4 L'exécution d'édition de surface de subdivision

L'édition de surface de subdivision multi-résolution peut être vue tout d'abord comme une subdivision récursive d'un maillage original, lequel génère de multiples maillages de surface aux différents niveaux de LOD, suivit de la manipulation de ces différents niveaux de LOD.

Lors de la manipulation de maillages de surface de subdivision, il est important de considérer deux éléments : le support local et la mémorisation

des changements effectués sur les points de contrôle à chaque niveau.

En ce qui à trait au procédé d'édition interactive, l'efficacité joue un rôle prépondérant. En fait, lorsqu'un sommet est modifié, ce changement influence seulement les sommets voisins. En observant les valeurs propres des matrices utilisées pour le calcul des points de contrôle aux niveaux plus élevés, nous remarquons que l'influence sur les sommets voisins dans le cas du schéma de *Loop* suit la règle de l'anneau simple. Une manière efficace de mettre à jour les maillages des niveaux supérieurs est donc de recalculer uniquement les sommets voisins influencés plutôt que de recalculer tout le maillage.

Une fois que les points de contrôle sur le maillage courant (lesquels peuvent être vus comme les parents des sommets du maillage de surface du niveau de subdivision supérieur) sont manipulés, les sommets des maillages des niveaux supérieurs (qui peuvent être vus comme les enfants des points de contrôle du maillage inférieur) vont être localement recalculés. Il en découle donc que lorsque les sommets parents sont modifiés, étant donné que les sommets enfants seront recalculés, les modifications antérieures des sommets enfants pourraient être perdues.

Nous proposons la solution suivante. Une fois un sommet édité, sa modification sera mémorisée dans une structure « Change » dans le maillage courant. Celle-ci contient l'index du sommet modifié ainsi que la différence entre son ancienne position et sa nouvelle. De plus, ces structures « Change » sont également présentes dans l'exécution de l'algorithme de support local.

Notre méthode consiste donc à emmagasiner tous les changements effectués pour ensuite les réappliquer lors de tout calcul ultérieur de subdivision totale ou locale. Ces changements peuvent être trouvés dans les « *AddNewChanges* » et « *ReApplySelectedChanges* ».

En ce qui concerne nos techniques d'édition, nous proposons deux types de création de plis sur les surfaces de subdivision de schéma de *Loop* : les plis sur les arêtes et les plis sur les sommets. Également, nous appliquerons le type de pli sur les arêtes aux surfaces de subdivision de *Modified Butterfly*. Les plis sur les sommets servent à garder certains sommets totalement fixes durant l'opération de subdivision. Les plis sur les arêtes servent à définir certaines arêtes intérieures comme étant des arêtes de bords. Concernant ces plis, nous avons ajouté certaines nouvelles règles pour les cas spéciaux.

4. Évaluation et résultats

Notre évaluation débute par l'analyse du temps requis pour les calculs de connectivité, les normales de surface et l'exécution du schéma de subdivision, le tout pour différents niveaux de détails. Par la suite, nous avons évalué la différence entre le calcul complet et le support local en utilisant le schéma de subdivision de *Loop*. Aussi, nous avons effectué différentes manipulations sur des maillages de surface pour prouver la validité de l'algorithme de conservation des changements de niveau supérieur vers un niveau inférieur. Finalement, nous démontrons les résultats obtenus en utilisant les plis sur des maillages arbitraires.

De l'évaluation de notre environnement de subdivision multi-résolution, nous observons que chaque fois qu'une subdivision, de *Loop* ou de *Modified Butterfly*, est exécutée, le nombre de sommet sur le maillage de surface augmente, ce qui détermine directement, principalement et linéairement le temps de calcul des algorithmes de subdivision, de connectivité de maillage et des normales de surface. En outre, nous sommes en mesure de conclure que le choix entre les schémas de *Loop* et de *Modified Butterfly* n'est pas un facteur qui influence significativement le temps de calcul global de la subdivision.

En ce qui concerne la validité de l'exécution du support local, nous avons comparé les différences de temps de calcul avec et sans support local pour le schéma de subdivision de *Loop*. Les résultats démontrent de manière évidente que l'utilisation du support local est plus efficace. D'ailleurs, nous pouvons conclure que plus le niveau de subdivision, la complexité et la densité des maillages augmentent, plus le temps de calcul économisé augmente lui aussi.

Nous avons prouvé la validité de l'algorithme de conservation des changements des niveaux supérieurs en illustrant le processus d'édition interactive sur des surfaces de subdivision de schéma de *Loop*.

Finalement, nous avons montré, avec les résultats de l'exécution des plis sur les arêtes et sur les sommets pour les schémas de *Loop* et de *Modified Butterfly*, qu'il est possible de garder des caractéristiques de plis sur des

surfaces de subdivision.

5. Conclusion et travaux futurs

Au début de ce mémoire, nous avons fourni une liste détaillée de travaux connexes et nous les avons classés en quatre sujets : représentations de surface, surfaces de subdivision, les techniques de multi-résolution et LOD et l'édition interactive de maillages. Nous avons décrit les théories fondamentales, présenté leurs utilités dans certains domaines, discuté de leur difficulté et relevé leurs innovations et approches pertinentes. Nous avons présenté un survol complet des références importantes, afin d'éclairer nos recherches.

Par la suite, nous avons implanté un schéma de subdivision d'approximation, *Loop*, et un schéma de subdivision d'interpolation, *Modified Butterfly*, afin de les utiliser comme schéma de base pour générer des maillages de surface de subdivision. Nos travaux se distinguent des autres, car ils fournissent une comparaison approfondie entre ces deux schémas de subdivision et démontrent comment les utiliser ensemble de manière efficace.

Ce que notre recherche amène de particulier est la possibilité de créer deux sortes de plis sur les deux schémas de subdivision. D'une part, nous avons mis en place deux sortes de plis pour le schéma de *Loop*. Pour ces dernières, nous avons ajouté des règles de calcul spécifiques pour les cas où deux plis ou plus se rejoignent. D'autre part, nous avons mis à jour certains masques de

calcul de bord et de plis pour le schéma de subdivision de *Modified Butterfly*.

Afin de rehausser les spécifications intuitives des surfaces de subdivision, nous avons amélioré l'algorithme de notre système d'édition interactif. Ainsi, une autre innovation de notre recherche se trouve dans le développement d'un algorithme résolvant les problèmes liés à la mémorisation des changements sur les points de contrôle des surfaces de subdivision sur les niveaux supérieurs tout en garantissant le support local.

Et finalement, nous avons ajouté de nouvelles fonctionnalités pratiques afin d'évaluer les performances des systèmes. Grace à ces fonctionnalités, nous avons obtenu des statistiques concrètes que nous avons pu utiliser afin de vérifier la validité de nos assertions.

Suite à tout ce que nous avons vu, il est facile de constater que les domaines de la subdivision et de la multi-résolution sont en pleine effervescence. Lors de l'exécution de notre projet, plusieurs idées nous sont venues à l'esprit. Celles-ci pourront éventuellement donner lieu à des recherches plus approfondies, par exemple, la subdivision adaptative permettrait de produire des maillages de résolution non-uniforme. Cela donnerait la possibilité d'obtenir plus de détails sur certains endroits d'un maillage. Autre exemple, suite aux résultats du chapitre quatre, il est facile de voir que l'amélioration des algorithmes de connectivité accélèrerait grandement la vitesse de calcul des surfaces de subdivision. Enfin, l'évaluation de la qualité des surfaces, qui est un domaine présentement actif, pourrait faire partie intégrante d'une suite de notre projet.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

2D/3D/4D	Two/Three/Four dimensional
ADM	Attribute Deviation Metric
AIF	Adjacency and Incidence Framework
B-Spline	Bésier Spline
CAD	Computer Aided Design
CAGD	Computer Aided Geometric Design
CAVE	Cave Automatic Virtual Environment
CPU	Central Processing Unit
CSG	Constructive Solid Geometry
DGP	Digital Geometry Processing
E	Edge
F	Face
GI	Graphics Interface
GIS	Geographic Information System
GLUT	OpenGL Utility Toolkit
GPU	Graphics Processing Unit
JADE	Just Another DEcimator
L2	Level 2
LOD	Level of Detail
MATLAB	MATrix LABoratory
M.E.S.H	Measuring Error between Surfaces using the Haudorff distance

MT	Multi-Triangulation
NURBS	Non Uniform Rational B-Splines
OpenGL	Open Graphics Language
PC	Personal Computer
PEL	Point-to-an-Edge-List
QEM	Quadric Error Metric
R^d	Region in dimensional space
RAM	Random Access Memory
TLDS	Triangular-Loop Data Structure
V	Vertex
WEDS	Winged-Edge Data Structure

LIST OF APPENDIXES

CHAPTER 1

INTRODUCTION

"I hear and I forget. I see and I remember. I do and I understand."

– Chinese Proverb

Surface meshes are widely used for simulation, visualization and optimization in modeling of physical objects. Their applications extend through industrial design and manufacturing, physiological organisms for surgical planning, mental models for the interpretation of ultrasonic images, and physical phenomena such as effect or force exerted by varying electrical and magnetic fields. These applications have already become useful predictive and cognitive tools incorporated into geometric modeling. However, even though much advancement has been made in this domain, the challenge to handle complex surface meshes consisting of thousands of triangles in an interactive environment always places a huge burden on modeling and rendering systems. This performance issue has been a preoccupation in geometric modeling almost as long as it has existed. Accompanied with the issue of maintaining a balance between quality and speed, researches on improving methods of editing surface meshes efficiently, play a significant role in the geometric modeling process. Our project aims to investigate multi-resolution surface mesh editing approaches through the development of a subdivision-based framework. Our approach to modeling should allow users to interact with subdivision-based surface meshes in a multi-level

environment.

The introduction draws the outline of our research in three major parts: an overview of the geometric modeling process, the motivation and the objective, the organization of interactive surface editing framework and a general structure of the thesis.

1.1 An Overview of the Geometric Modeling Process

Our world exists in three-dimensional space and it develops in a four-dimensional environment that involves physical space and time. What we can see and feel is in 3D all the time. We make use of geometric modeling to describe and capture a wide range of spatial positioning and placement information of virtual objects in 3D. Most of the time, we take the modeling of our reality for granted, however, when it comes to put this reality into a visualizing and editing environment using computer programming, we have to follow various steps in a process that has become pretty much standardized.

In order to visualize and manipulate 3D objects interactively, models are created through a geometric modeling process, which comprises the following five steps: (1) setting world coordinates to locate, place and move 3D geometric models; (2) generating numerical descriptions of geometric models

by using proper representations; (3) building a modeling system to modify the original models by scaling, moving and editing; (4) optimizing or simplifying the models to extract specific details from the approximating models according to evaluation criteria, such as memory requirements and reconstructing rules; (5) implementing the final models for diagnosis, analysis or different applications. Schematically, the general geometric modeling process with its five steps can be illustrated as Figure1-1.

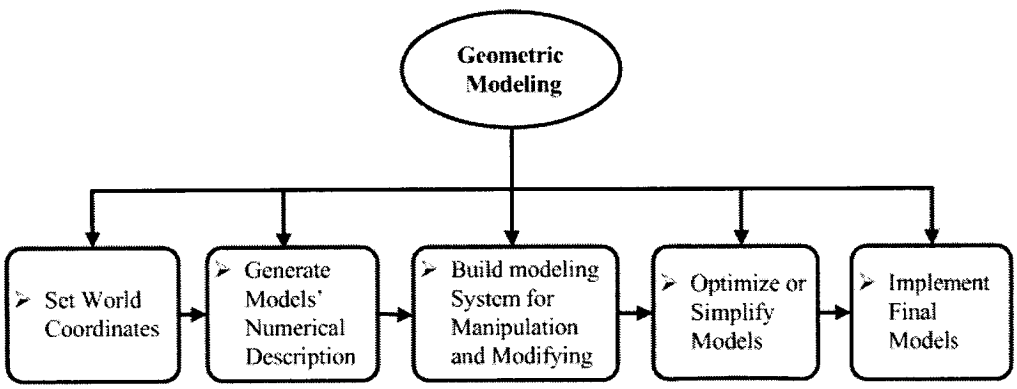


Figure 1-1: The general geometric modeling process.

The first step is to set a 3D environment for modeling. It is based on the simple principle of world coordinates. All computer modeling systems and 3D applications rely on coordinate systems to represent the 3D world. In 3D graphics, the Cartesian coordinate system with XYZ representation is used in most geometric modeling applications (see Figure 1-2). Most of the time, according to the definition of planes, which are paired by any two axes, we can display models from different points of view (see Figure 1-3).

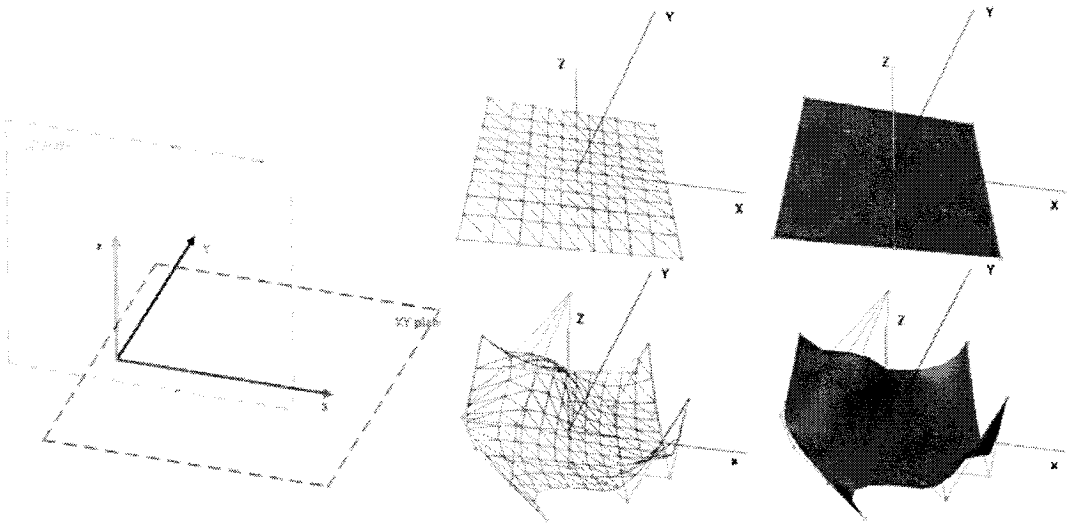


Figure 1-2: The Cartesian coordinate system for building models.

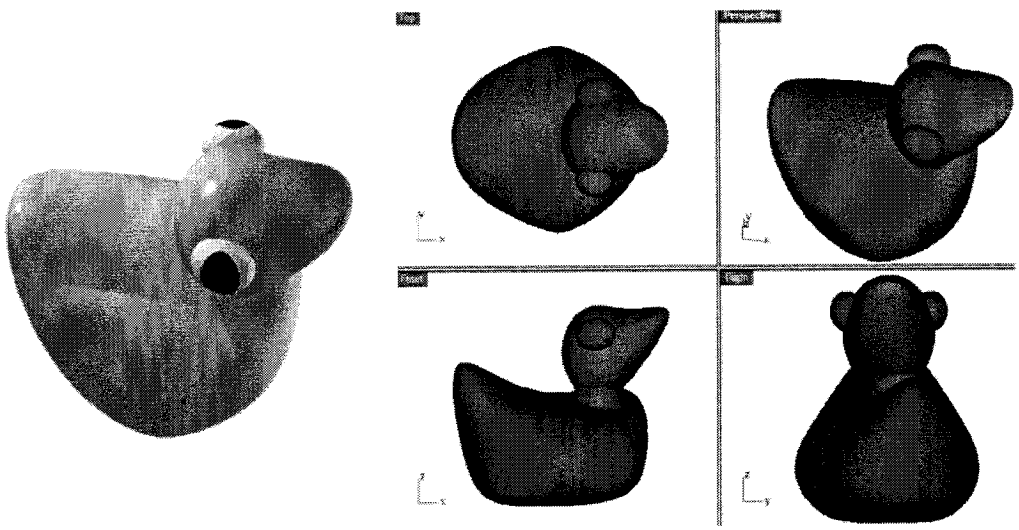


Figure 1-3: Visualizing the model of a duck from different points of view.

Geometric modeling needs very precise descriptions of objects' shape, including their specific positions in 3D world. When modeling comes with computer programming, we need to understand how a specific computer

system defines objects using numerical descriptions.

Most numerical descriptions originate from the combination of unorganized points. Once all these points are interpreted into sets of 3D vertices and interconnected into triangles, meshes are used to visibly outline the edges connecting the vertices. In computer graphics, a mesh is a general representation of 3D objects. The underlying design structure of 3D models can be visualized using meshes (see Figure 1-4). Since mesh rendering is relatively efficient in terms of computational cost, meshes are widely used to display and handle complex 3D models in most modeling systems (see Figure 1-5).

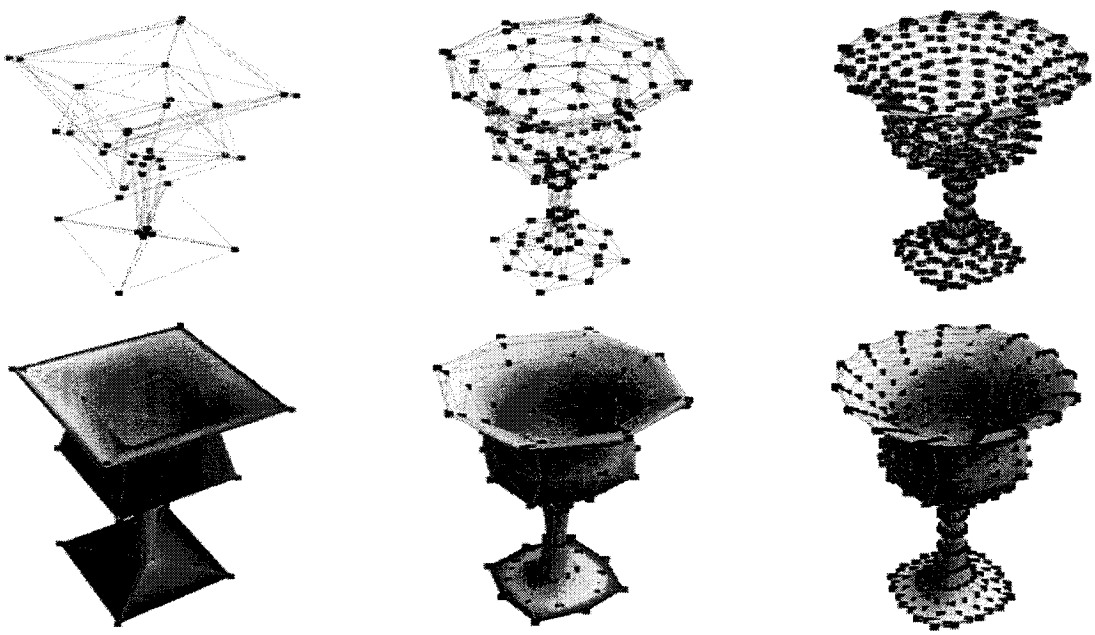


Figure 1-4: Model cup meshes displaying with wire-frame mode.

The original sources of meshes are the combinations of the objects' details. On one side, they can be built through iterative refinement of basic primitives

controlled by their types and dimensions. In this way, models are constructed using modeling tools, such as Maya (Alias®), Rhinoceros®-Nurbs Modeling for Windows, ZBrush (©Pixologic) or 3D Studio Max (Autodesk®). On the other side, a variety of equipments, such as 3D laser scanners, digital cameras, and laser rangefinders are useful solutions to obtain accurate data of objects.

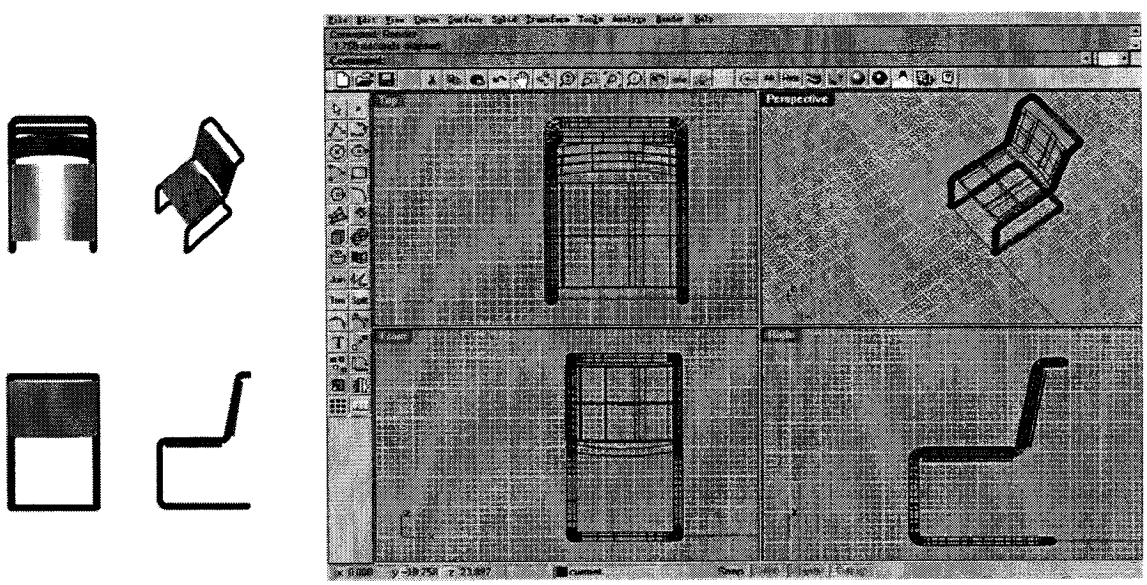


Figure 1-5: Model chair rendering in the modeling system (Rhinoceros).

The essential problem in this second step involves choosing appropriate mesh representations to work with. Models' surfaces can be represented in various ways. Polygon meshes, which can be used to describe any arbitrary topology, are commonly used in modeling. Polynomial patches, such as NURBS (Non-Uniform Rational B-Spline) and Bézier patches, offer an alternative approach to construct models' surfaces, using relatively small numbers of control parameters. Both of them have their strongpoint and shortcoming, and one of them is possibly better suited for certain application, while the

other performs better in different contexts. For example, polygon meshes are not limited in their descriptions of any arbitrary topology at any level of detail, whereas the complexity of meshes can increase prohibitively. Once we interact with complex polygon meshes, current hardware system will have difficulty in coping with this huge burden. Different from polygon meshes, since improved patch algorithms are available, polynomial patches can be implemented without over-loading the hardware system. But as polynomial patches describe finer details, the cost to calculate patches' control parameters will be more expensive.

In our project, we aim to investigate and implement subdivision surfaces, which fall somewhere between polygons' arbitrary topology and piecewise patches' independent resolution, as the foundation of a hierarchical polyhedral mesh representation. What we need to emphasize is that subdivision surface combines the merits of other arbitrary topology, handling large-scale manipulation of model details, and supporting well multi-level editing. In reverse, other representations have a hard time matching these important points.

Once a model's numerical descriptions is generated, we can render or animate it through computer programming. Now there is another important step, model editing, which must meet various design requirements. In this step, specific modeling methodologies have to be followed. While basic geometric structure and details of models are preserved, model editing involves a variety of functions: transforming and scaling, cutting and implementing parts of models, interacting with models in free-form

deformation, handling geometric representation by moving its control points and refining models.

Editing surface meshes with different levels of details is essential in interactive computer graphics. However, when level of resolution goes higher, it means that interactive display and rendering will be slower, which affects system performance and makes models more difficult to work with. That's why, on one side, the editing process brings the requirements of performance and fidelity to the manipulation of models representation. On the other side, it is important to improve the flexibility and realism of the geometric modeling process. However, it is not easy to incorporate or harmonize all these factors at the same time. Thus, the editing process surely meets some crucial challenges, especially the improvement of data structures and algorithms. This is one of the key aspects that will be investigated in the present research.

When the editing step is completed, efficiently transferring and rendering models, also becomes a necessary step. Sometimes, models are composed of too much redundant information. We can optimize models by deleting unnecessary information, while still keeping the main features. This simplifying and optimizing process must meet the constraints imposed by target applications, which are two-fold: fidelity and triangle-budget. Various research contributions on simplification and optimization processes according to local and global simplification operators have already been approached. Our work should directly benefit from this research.

Finally, the refined models can be implemented in various run-time environments. This step is not the last stage, and researches on this step will possibly contain other advanced issues. In our thesis, we only outline this part of the modeling process.

1.2 Motivation and Objective

After examining the process of geometric modeling, we can summarize the existing problem in this process as: how to efficiently edit 3D models at various levels of details? As we know, the ultimate goal of geometric modeling is to produce vivid models with plasticity and authenticity. This is a formidable challenge to dynamically render models with high level of details. For interactive manipulation of models' surface meshes, this challenge is aggravated by the need to effectively interact with models at different levels of details. Thus, the overall challenge appropriately motivates us to explore the existing concepts, algorithms and data structures in depth. Moreover, it inspires us to compare theoretical algorithms and implement practical data structures in order to contribute to research in this domain.

In particular, the motivation of this research is sparked by simple and efficient subdivision modeling, which has already been applied in a lot of different fields, particularly for industry product modeling and video game animation. Subdivision schemes appear as an efficient way of smoothing complex objects' surfaces, thus it is interesting and useful to take time to

investigate the topic of subdivision schemes.

Furthermore the recent advances on multi-resolution in geometric modeling have reinforced the development of efficient data structures for modifying the control points of subdivision surface meshes at different levels. Level-Of-Detail (LOD) criteria are proposed in the framework of multi-resolution meshes to represent 3D surfaces.

In essence, all these editing issues embrace not only the methods for manipulating control points on surface meshes but also the design constraints that enable us to interface with modeling and obtain an efficient rendering at different levels. In order to find solutions to problems related to the interactive manipulation of surface meshes, our research extends from pure subdivision schemes to the category of multi-resolution and level-of-detail framework.

The objective of this research is to design a multi-resolution subdivision-based framework for interactive surface editing, which allows users to efficiently interact with multi-resolution subdivision surfaces. The interactive surface editing framework must satisfy the following requirements:

1. This framework must implement effective algorithms and data structures to keep a balance between system speed and graphics fidelity, which will permit manipulating, modifying and efficient rendering of subdivision-based representations at different levels of details;

2. This framework must propose an extensible and a convenient GUI (Graphical User Interface), which will well preserve required details and allow to implement and test interactive editing strategies;

3. This framework must implement important subdivision features, used for geometric modeling, such as the handling of creases, which will put new insight into the future development of subdivision technology.

1.3 The Organization of Framework and Thesis

Our subdivision-based framework is organized in three interdependent contents: subdivision schemes, multi-resolution technology and interactive editing functions. All these contents are based upon 3D surface mesh modeling and the content of interactive surface editing will be emphasized.

We implement several mesh-handling functions: smoothing, selecting, dragging and definition of creases. The 3D subdivision surface meshes can be rendered with level of details and mesh modifications can be performed interactively. The full description of the functions implemented is presented in Appendix II.

The thesis is organized as follows. Chapter 2 "Related Work" provides references for basic concepts of surface representations, subdivision surfaces,

technologies on multi-resolution and LOD. In this chapter, surface representations and subdivision schemes are introduced and classified. Moreover, surveys of representative applications of subdivision surface, multi-resolution technology and LOD, vital techniques of interactive mesh editing, are addressed.

In order to cover both theory and application, Chapter 3 "Implementation of Multi-resolution Surface Mesh Editing" describes techniques used for generating multi-resolution surface meshes and for manipulating mesh at multiple subdivision levels. The focus of this chapter is to implement the Loop scheme and the Modified Butterfly scheme and give a detailed comparison between the approximating scheme and the interpolating scheme. Meanwhile, we select concrete methods for editing surface meshes to present the multi-resolution subdivision-based framework. At the end of this chapter, we present the data structures and algorithms with local support, while keeping all changes from different levels.

In Chapter 4 "Evaluation and Results", the performance of our project is evaluated using statistics of time costs arising from performing specific real-time tasks. With respect to the objective of our project, the relevant results are analyzed. We describe the concrete environment of system evaluation including its Graphical User Interface in Appendix II.

Chapter 5 "Conclusion and Future Work" contains a summary of our work and several forward-looking issues on subdivision, multi-resolution technology, editing methods and surface quality evaluation.

CHAPTER 2

RELATED WORK

In this chapter, we place our research within the larger context of related work on surface representations, subdivision surfaces, multi-resolution technique with LOD (level of detail) and interactive mesh editing. In section 2.1, we describe a concise classification of common surface representations and establish geometric fundamentals for the implementation of subdivision surfaces in our framework. In section 2.2, we introduce the definition of subdivision surfaces, supply an overview of subdivision schemes, a variety of practical applications of subdivision surfaces, extending from engineering design to scientific and medical visualization, and indicate the motivation of implementing subdivision surfaces in our project with a brief comparison of common surface representations. In section 2.3, we present a survey of relevant applications of multi-resolution meshes and advanced issues on Level-Of-Detail (LOD) of a real-time multi-resolution framework. In section 2.4, we investigate approaches related to the interactive editing of multi-resolution meshes. After the discussion of the relevant works, we restate the novelty and the contribution of our research in section 2.5.

2.1 Surface Representations

Surface representations are essential components of any geometric modeling system. In this section, we begin with the introduction of common geometric representations and then move on to describe some of the most commonly used surface representations.

2.1.1 Common Geometric Representations

The representations of 3D geometry provide the foundations for geometric modeling. Considering the respective properties and specific features of 3D objects, they are represented in different types of representations with distinct data structures and unique operations. The 3D object representations in common use are classified concisely in Figure 2-1 [Naylor et al. 96, Funkhouser 03, Guibault 03, Finkelstein 05].

In 1996, Bruce Naylor gave a partial taxonomy of representations of geometry (see Figure 2-2) [Naylor et al. 96]. He also brought an important consideration in geometric computation: the verbosity/complexity trade-off (see Figure 2-3), which is described as a relatively small number of complicated operations that can be traded for many more but simpler operations. This consideration enlightens more perceptual issues on computational differences: efficiency, simplicity and usability. And these computational differences can be used to assist researchers in measuring and

analyzing research results.

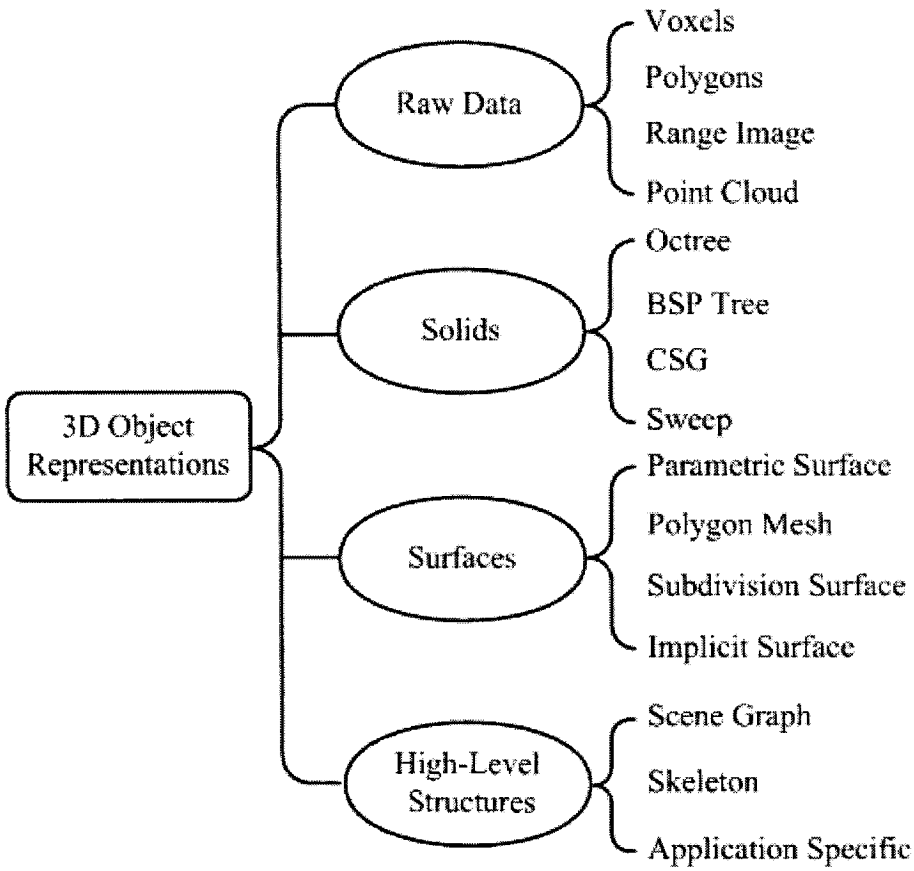


Figure 2-1: A concise classification of geometric representations.

Each of these fundamental geometric representations has sufficient expressive ability to model the features of any definite object. Moreover, each of them possesses a distinct set of operations to manipulate the properties of an object. The surface representations most commonly in use are polygon meshes, parametric surfaces, implicit surfaces and subdivision surfaces. In the following part of this section, the first three kinds of surfaces will be given a general description and an analysis about their pros and cons. Subdivision

surface will be introduced in detail in section 2.2.

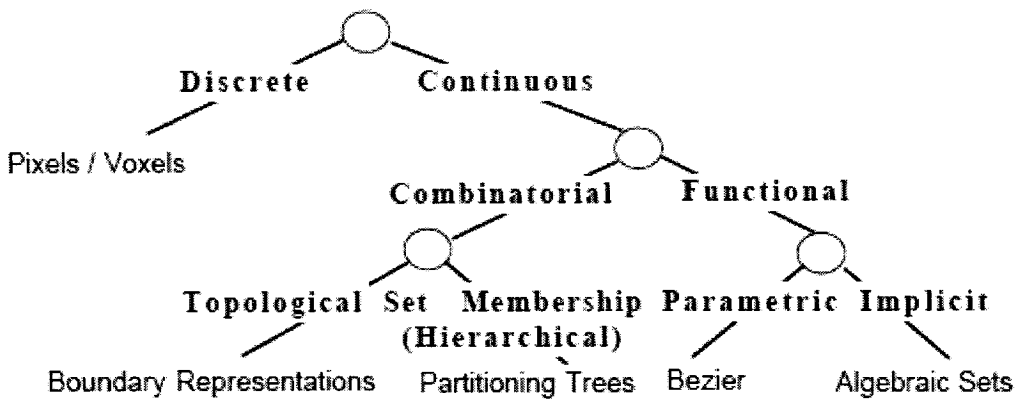


Figure 2-2: The partial taxonomy of representations of geometry [Naylor et al. 96]. Copyright © SIGGRAPH 1996.

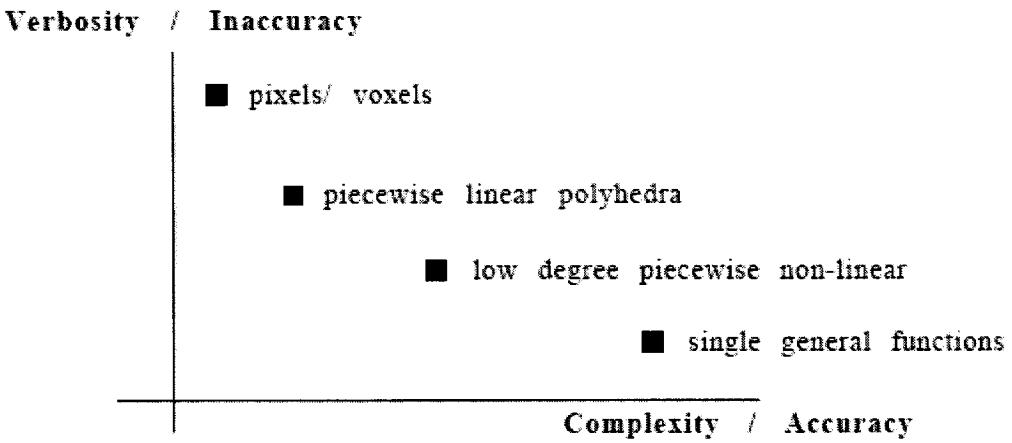


Figure 2-3: Verbosity/Complexity trade-off for geometric representations [Naylor et al. 96]. Copyright © SIGGRAPH 1996.

2.1.2 Implicit Surfaces

In geometric modeling, implicit equations and parametric functions are the two most common methods to represent surfaces [Piegl et al. 97, Hodgins 01, Guibault 03]. The implicit surface in the 3D Euclidean space can be viewed as a set of points that satisfy $f(x, y, z) = 0$; while the other closed points bounded by the surface that satisfy $f(x, y, z) < 0$ define a solid. The implicit method is particularly well suited to describe molecules, to calculate the surface intersection in relation with collision detection, and to test points inside or outside boundaries. An example of implicit surfaces is a quadric surface, which is applied in ray-tracing technology [Goldman 83]. The implicit form of quadric surfaces is given by Equation 2.1:

$$f(x,y,z) = ax^2 + by^2 + cz^2 + 2dxy + 2eyz + 2fxz + 2gx + 2hy + 2jz + k = 0 \tag{2.1}$$

The shortcomings of implicit surfaces include cumbersomeness to represent surface patches and non intuitiveness to represent surface information with orientation. Correspondingly, parametric surfaces using parametric functions are more natural for representing oriented surfaces with bounded curve segments [Piegl 97].

2.1.3 Parametric Surfaces

The surface in 3D is described using the parametric function with the parameters *u, v* as Equation 2.2:

$$\begin{cases} x = f_x(u,v) \\ y = f_y(u,v) \\ z = f_z(u,v) \end{cases} \quad u,v \in R \ (R \text{ is a region}) \tag{2.2}$$

For parametric surfaces, the boundaries of an object (see Figure 2-4) are partitioned into parametric patches, where each patch relies on blending of control points *P(x, y, z)*. The points on any patch depend on blending functions which are generally defined by polynomials. Using a parametric representation, the *Q(u, v)* is expressed as in Equation 2.3, where *M* is a matrix describing the blending functions for a parametric cubic curve [Piegl 97, Funkhouser 03, Guibault 03, Finkelstein 05]. It is the different values of *M* that result in the different properties of surfaces.

The polynomial blending functions play an important role in extending the flexibility to represent different kinds of parametric surfaces. As the degree of polynomials is raised, the flexibility in controlling the shape of surface increases, while the cost of computations increases and unexpected wiggles become possible.

$$Q(u,v) = UM\begin{bmatrix} P_{1,1}P_{1,2}P_{1,3}P_{1,4} \\ P_{2,1}P_{2,2}P_{2,3}P_{2,4} \\ P_{3,1}P_{3,2}P_{3,3}P_{3,4} \\ P_{4,1}P_{4,2}P_{4,3}P_{4,4} \end{bmatrix} M^T V^T \qquad (2.3)$$

$$U = [u^3 u^2 u^1 1]$$

$$V = [v^3 v^2 v^1 1]$$

Different polynomials are used to represent surfaces with variable specifications. For example, Bernstein/Bézier polynomials are used for the polynomial blending function to represent Bézier surfaces. Rational B-Spline basis functions are implemented as NURBS (Non-Uniform Rational B-Splines).

Parametric surfaces are powerful representations for the construction of complex objects. Polynomial patches endow parametric surfaces with an exclusively geometric significance in constructing models using relatively few control parameters. As a result, their pros include that it is easy to manipulate them interactively and reconstruct them by interpolation and approximation; representing them is not cumbersome by enumerating points on surfaces; and their ability in economizing memory is efficient. Their cons include that the control mesh must be quadrilaterals; continuity constraints are difficult to maintain; it is hard to find intersections and representing arbitrary topologies is a big challenge [Zorin et al. 97].

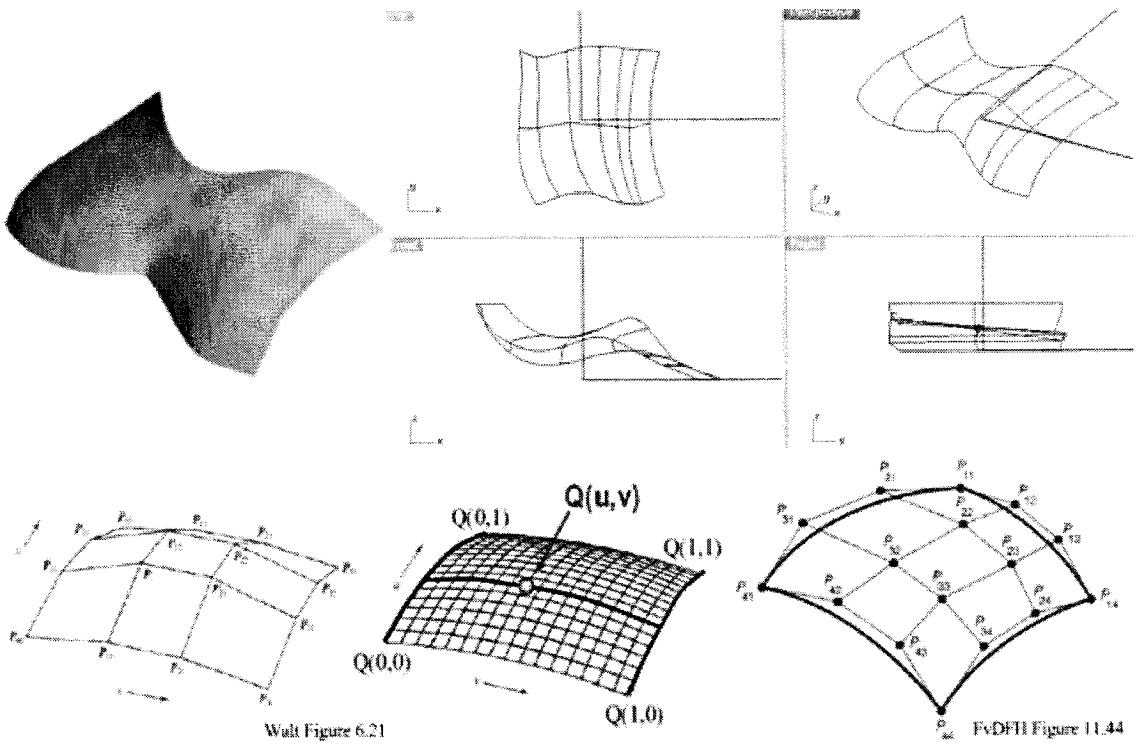


Figure 2-4: Parametric surface of parametric cubic curves [Finkelstein 05].

2.1.4 Polygon Meshes

Objects can also be represented by a large collection of polygons. A polygon mesh is constructed from three basic elements: vertices, edges and faces [Naylor et al. 96, Hodgins 01, Guibault 03]. Each vertex is the conjoint point for edges, which are the boundaries between two faces. In brief, the relations among vertex, edge and face are listed as the following:

- Vertices (V): Points;
- Edges (E): Line segments between two vertices;
- Faces (F): Polygon bounded with edges;

Any polygon mesh follows Euler formula and satisfies Equation 2.4.

$$V - E + F = 2 \tag{2.4}$$

Polygon meshes are collections of connected planar polygonal surfaces. The representative example is a triangle mesh, such as the one illustrated in Figure 2-5. Because of their algorithmic simplicity, numerical robustness and efficient rendering, polygon meshes, especially triangle meshes, are widely used in computer graphics applications [Kobbelt et al. 00].

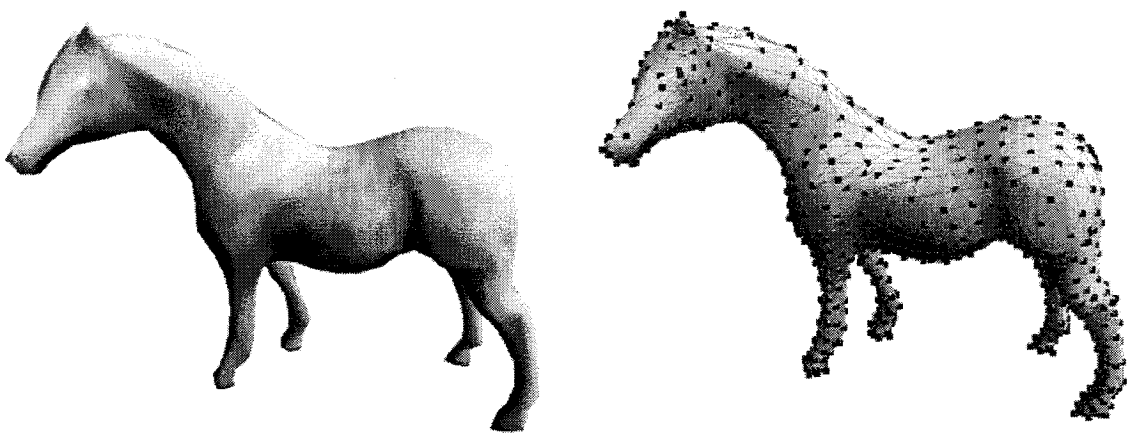


Figure 2-5: The model horse is rendered using triangle meshes.

The computational complexity of polygon meshes depends on the number of polygons. When more polygons are used, the details of models become more accurate while the cost in memory and time for executing and rendering increases. So an optimal choice of a polygonal model is to model polygon meshes with the minimum number of polygons while meeting the maximum

complexity requirements. *Refinement* and *decimation* methods [Hoppe 96, Kobbelt et al. 00] are used to meet this trade-off.

Considering the mesh geometry and the mesh connectivity, the researches about mesh simplification viewed as an optimization process under constraints, such as fidelity-based and triangle-budget-based, are improved by the approaches of global or local simplification operators [Luebke et al. 02]. The global simplification operators include volume processing (ex. low-pass filtering and morphological operators) and Alpha-Hull-Based topology simplifications. The local simplification operators include edge collapse [Hoppe et al. 93, Hoppe 97, Bajaj et al. 99] (ex. foldover and topological inconsistency), vertex-pair collapse [Popovic and Hoppe 97, Schroeder 97], triangle collapse, cell collapse, vertex removal, polygon merging and general geometric replacement. There are a variety of commercial or academic software packages for mesh simplification available, whose performances are selectively compared in the survey written by Surazhsky and Gotsman [Surazhsky and Gotsman 05].

In order to obtain favourable results, it is crucial to measure errors between original meshes and final meshes during and after the simplification process. The error measurement imposes a significant impact on the quality of reconstructed results. Thus the issues on quality assessment discussed in *Comparison of Mesh Simplification Algorithms* [Cignoni et al. 98] are uniquely referential. The related approaches on simplification error metrics [Luebke et al. 02] include local geometric errors such as the calculation of the vertex-plane distance with the representative example: QEM (Quadric Error

Metric) [Garland et al. 97, Heckbert et al. 98, Heckbert et al. 99, Garland 99-Ph.D. thesis, Wu et al. 04, Garland et al. 05], the surface-surface distance (to calculate maximum, mean and mean square errors between two surfaces) using Hausdorff distance [Aspert et al. 02], and the vertex-surface distance (ex. ADM (Attribute Deviation Metric) [Roy, Foufou et al. 02, Roy, Nicolier et al. 02, Roy, Foufou et al. 04]); and global error measurements with both numerical and visual results [Cignoni et al. June 98]. In addition, software programs for measuring the distortion between two given surface meshes such as M.E.S.H. (Measuring Error between Surfaces using the Hausdorff distance) [Aspert et al. 02], QSlim [Garland 99-Ph.D. thesis], Metro [Cignoni et al. June 98], ProgMesh [ProgMesh © Hoppe] and MeshDev [Foufou et al. 04, Roy, Foufou et al. 02, Roy, Nicolier et al. 02, Roy] are publicly available to determine surface mesh quality.

In brief, polygon meshes can represent any arbitrary topology while achieving fine details. Polygon meshes are an optimal choice as the basic primitives to render models in geometric modeling. However, arising from tremendous requirements for preserving the levels of detail during interactive editing of polygon meshes, the huge amounts of detail data make current hardware difficult to support handling or manipulations in real-time applications. In a certain way, the solutions for preserving fine details at a lower cost can be reckoned on the processes of mesh simplification [Hoppe et al. 93] and subdivision surface with hierarchical details [Zorin et al. 97].

2.2 Related Work on Subdivision Surfaces

In 1974, Chaikin introduced the basic paradigm of subdivision: generating curves from a limited number of points, known as control points, by utilizing the corner cutting algorithm [Chaikin 74]. This innovative introduction has brought the emergence of many different subdivision schemes. The beginning of subdivision for surface modeling was indicated by two pioneering schemes: the cubic Catmull-Clark subdivision surface [Catmull et al. 78] and the quadratic Doo-Sabin subdivision surface [Doo et al. 78]. In 1987, Loop proposed smoothing triangle surfaces using a subdivision method [Loop 87]. Later, with their properties as the foundation, subdivision surfaces were proposed for interactive multi-level editing and real-time animation [Zorin et al. 00]. Recently, subdivision surfaces have become a basic building operation in modeling software and a core technology in computer graphics.

2.2.1 Definition of Subdivision Surfaces

Subdivision surfaces in geometric modeling are a valuable modeling method that generates a complex smooth geometric model as the limit of a sequence of successive refinement of faceted surface meshes [Zorin et al. 00]. The surface generation process is defined as a smoothing sequence, which refines surface meshes from a coarser level to a finer level. The refinement process is a recursive procedure following different rules or performing different operations to achieve various smoothing.

Before we move on the discussion of subdivision surface further, we list the notations that we are going to use subsequently as below:

M^j : global mesh surface at j^{th} level, where $M^j \in R^d$;

m_i^j : i^{th} local faceted mesh surface at j^{th} level, where $m_i^j \in M^j$;

S : global subdivision operator, where $\lim_{k\to\infty} s_{i,k} = S$;

$s_{i,k}$: local subdivision operator performing k times on i^{th} faceted mesh surface;

v_i^j : i^{th} vertex at j^{th} level, which is a vector;

L : global linear subdivision, where $\lim_{k\to\infty} l_{i,k} = L$;

$l_{i,k}$: local linear subdivision performing k times on i^{th} faceted mesh surface;

A : global averaging, $\lim_{k\to\infty} a_{i,k} = A$;

$a_{i,k}$: local averaging performing k times on i^{th} faceted mesh surface;

d : the offet vector, or the vertex difference between different levels;

We can use Equation 2.5 to define the smoothing process.

$$M^{j+1} = S \cdot M^j \quad (j \in N) \tag{2.5}$$

The smoothing process is illustrated in Figure 2-6, where M^0 in 3D represents an initial polygonal mesh consisting of source data such as: vertices, edges and faces, where S refers to the subdivision operator following the rules to linearly insert new vertices and to move or settle down former vertices in order to achieve some special effect. The fundamentals of S can be demonstrated as two operations acting on the surface mesh: linear

subdivision and averaging [Warren and Weimer 02].

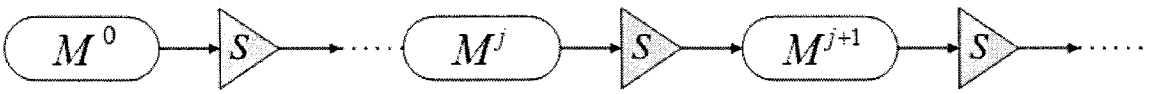


Figure 2-6: The illustration of the smoothing process using Equation 2.5.

We give a 2D curve consisting of vertices connected by edges as an example. Linear subdivision (Figure 2-7) performs insertion of new vertices in the middle of each edge. Averaging (Figure 2-8) performs a successively symmetrical shrinkage of the former vertices' positions.

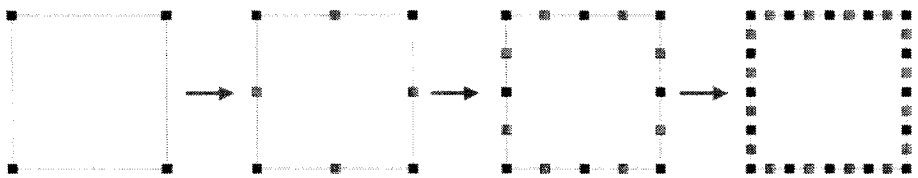


Figure 2-7: Linear subdivision (the new inserted vertices are illustrated with pink color).

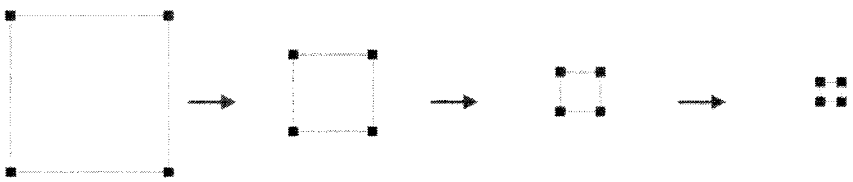


Figure 2-8: Averaging (the old retained vertices are shrunk along their diagonals).

Subdivision operator S is the combination of linear subdivision and averaging. Operator S can be defined to satisfy certain conditions, however it never goes

beyond the category of these two operations. Supposing M^0 still to be a curve in 2D, we can illustrate the process of S performed on a 2D curve in Figure 2-9. This example illustrates that the operator S executes a successive refinement of the original input and the final output is converged into a limited smooth result in sequence.

Replacing the 2D example with a coarse surface mesh in 3D, Figure 2-10 and Figure 2-11 illustrate the subdivision operator S being applied over 3D hierarchical polygonal meshes. Thus the definition of subdivision surface comes as: the limit of a sequence of successively refined polyhedral meshes [Zorin 98, Zorin et al. 00, Zorin et al. 97].

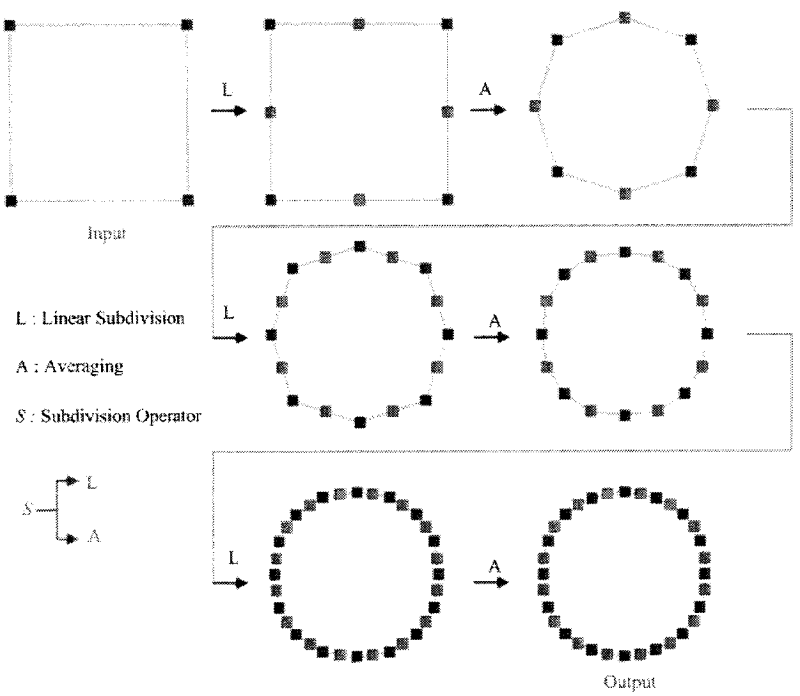


Figure 2-9: The subdivision operator S is performed over a 2D curve.

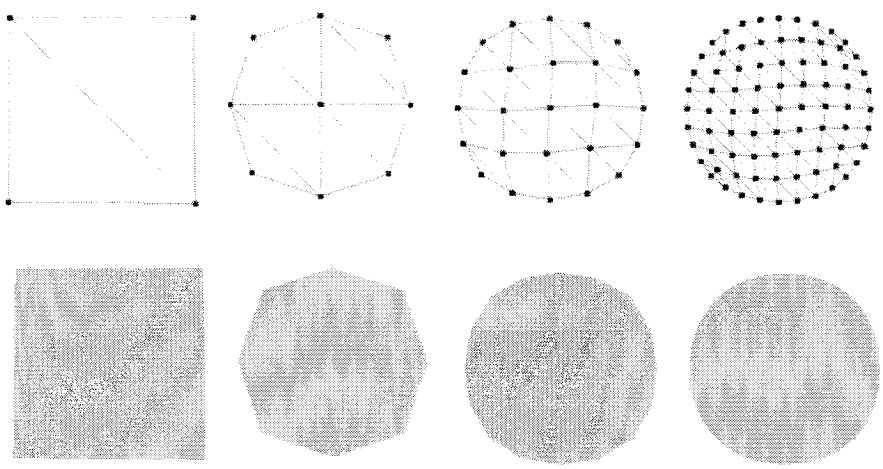


Figure 2-10: The subdivision operator S is performed over a 3D plane.

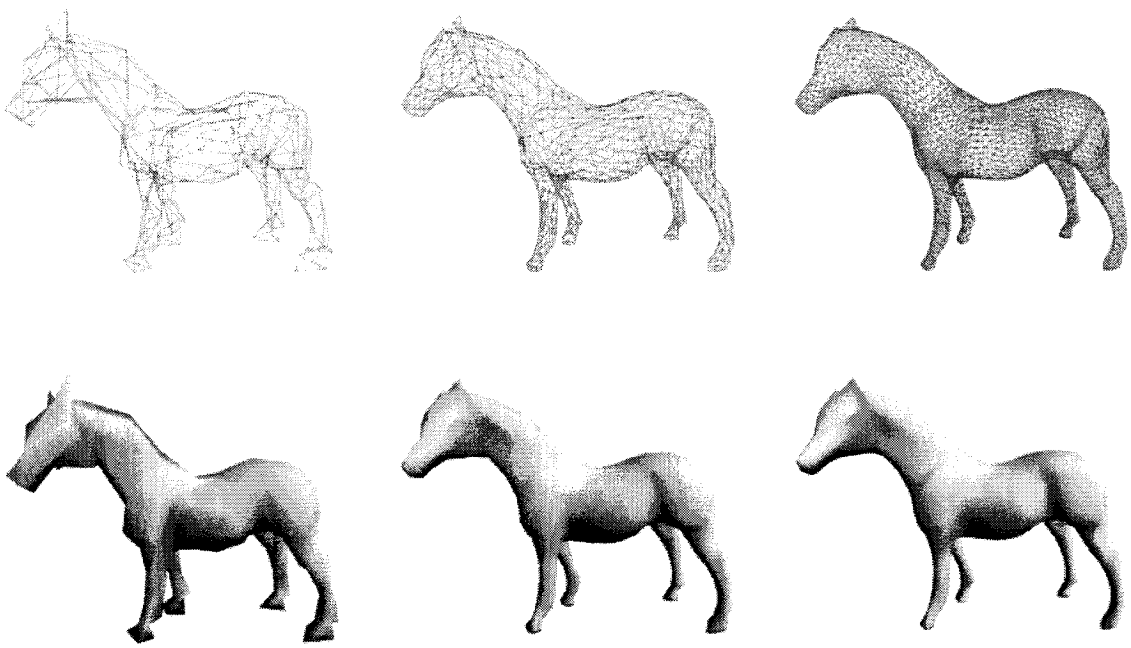


Figure 2-11: The subdivision operator S is performed over a horse surface mesh in 3D.

The process to perform the subdivision operator S as a linear combination of blended functions can be described as mapping the coarse mesh to the fine mesh finitely and linearly [Schröder 04]. Each faceted surface mesh including each vertex on it is mapped linearly to the surface mesh on another level in sequence or in reverse, which can be expressed by using Equation 2.6.

$$\begin{aligned} & v_i^j \longleftrightarrow v_i^{j+1}; \\ & M^j = \sum_{i=0}^{n} m_i^j \longleftrightarrow M^{j+1} = \sum_{i=0}^{n} m_i^{j+1}; \qquad (i,j,k \in N) \\ & M^{j+1} = S \cdot M^j \longleftrightarrow m_i^{j+1} = \sum_k s_{i,k} m_k^j \end{aligned} \tag{2.6}$$

The essential properties of subdivision operation S can be summarized as below [Zorin 98, Zorin et al. 00, Schröder 04]:

- **Compact Support**

  The influence of the region over which an original control point poses should be limited to the shape of the final surface meshes (see Equation 2.7);

$$\exists\, l_0, l_1 \in N \Rightarrow \forall k : s_{i,k} = 0, i \notin (2k - l_0, 2k + l_1); \qquad (i,k \in N) \tag{2.7}$$

- **Affine Invariance**

  The transformation of the original set of control point should result in the same object after subdivision as if the transformation had been applied to the subdivided object (see Equation 2.8);

$$a \cdot v_i^{j+1} + d = \sum_k s_{i,k}(a \cdot v_k^j + d); \quad (d \in vector, a \in R) \tag{2.8}$$

- **Index Invariance/Symmetry**

The matrix defining subdivision operator S should be indexed symmetrically (see Equation 2.9);

$$\begin{aligned} s_{i,k} &= s_{i+2,k+1}; \\ s_{2k-l,k} &= s_{2k+l,k}; \end{aligned} \qquad (i,k,l \in N) \tag{2.9}$$

- **Local Definition**

The rules of subdivision operator S used to determine the positions of new vertices should only depend on the local neighbour of its original control points (see Equation 2.10).

$$v_i^{j+1} = \sum_k s_{i-2k} \cdot v_k^j; \quad (k,i,j \in N) \tag{2.10}$$

2.2.2 An Overview of Subdivision Schemes

Subdivision schemes can be classified in two kinds: stationary and non-stationary. Most classical subdivision schemes are stationary. Because non-stationary subdivision schemes apply different subdivision rules at each refinement level dynamically and complicatedly, they are sorted out from stationary ones.

A general classification of various stationary subdivision schemes based on four criteria is presented by Zorin [Zorin 98, Zorin et al. 00]. This classification focuses on established subdivision schemes. The four criteria are:

1. The type of generated mesh, such as triangular, quads, regular hexagons;

2. The type of refinement rules: primal, dual and titling refinement;

(1). Primal (vertex insertion / face split)

- Triangles (see Figure 2-12)

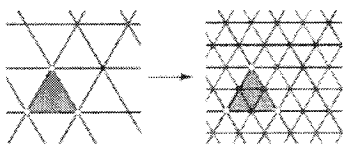


Figure 2-12: Face split for triangles [Zorin et al. 00].

- Quads (see Figure 2-13)

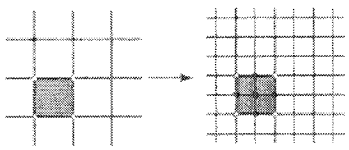


Figure 2-13: Face split for quads [Zorin et al. 00].

(2). Dual (corner cutting / vertex split)

- Quads (see Figure 2-14)

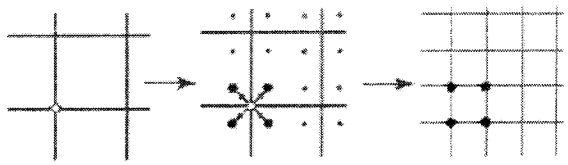


Figure 2-14: Vertex split for quads [Zorin et al. 00].

(3). Tilings refinement (isohedral / Laves)

- Honeycomb refinement (see Figure 2-15)

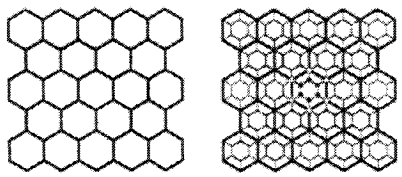


Figure 2-15: Honeycomb refinement [Zorin et al. 00].

- $\sqrt{3}$ refinement (see Figure 2-16)

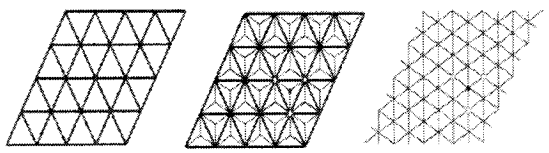


Figure 2-16: $\sqrt{3}$ refinement [Zorin et al. 00].

- Bisection (see Figure 2-17)

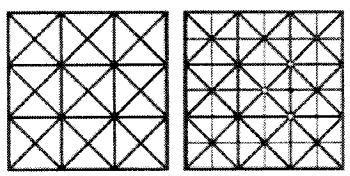


Figure 2-17: Bisection on a 4-8 tiling [Zorin et al. 00].

3. Whether the scheme is approximation or interpolation, depending on the way old control points from lower level are mapped onto points at higher level;

(1). Interpolation: the old control points on the original surface meshes

are retained on subdivided surface meshes as the former ones (see Figure 2-18);

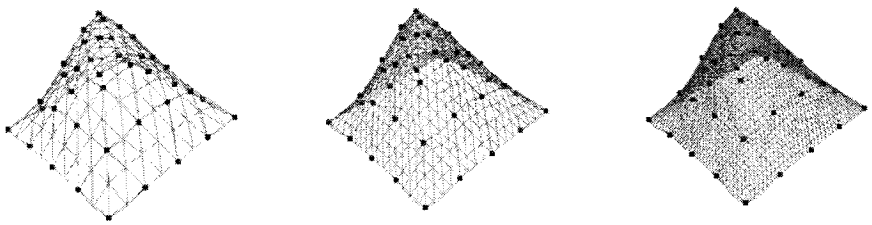


Figure 2-18: Modified Butterfly subdivision– 3D cone (blue points - old control points, pink lines – original control meshes).

(2). Approximation: the old control points on the original surface meshes are going to be changed on subdivided surface meshes (see Figure 2-19);

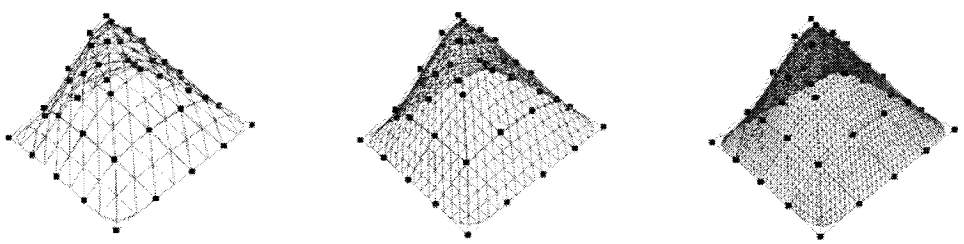


Figure 2-19: Loop subdivision – 3D cone (blue points - old control points, pink lines – original control meshes).

4. Smoothness / continuity of the limit surfaces for regular meshes C^n (ex. $C^1, C^2, C^3 \ldots$), where n^{th} derivatives or tangents are continuous.

The following table 2-1 [Guibault 03] summarizes this classification of

subdivision schemes:

Table 2-1: The classification of subdivision schemes is based on four criteria [Guibault 03].

Subdivision Scheme	Type of Mesh	Type of Refinement Rule	Type of Schemes	Continuity
Catmull-Clark	Quadrilateral	Primal	Approximation	C^2
Loop	Triangle	Primal	Approximation	C^2
Butterfly	Triangle	Primal	Interpolation	C^1
Kobbelt	Quadrilateral	Primal	Interpolation	C^1
Doo-Sabin	Quadrilateral	Dual	Approximation	C^1
Mid-Edge	Quadrilateral	Dual	Approximation	C^1
Bi-Quartic	Quadrilateral	Dual	Approximation	C^2
$\sqrt{3}$	6^3	Tiling	Approximation	C^2
4-8	4.8^2	Tiling	Approximation	C^4

2.2.3 An Overview of Subdivision Surfaces' applications

To enhance the further discussion of subdivision surfaces' features, in this section we look at related works on subdivision surfaces' applications in three distinct domains.

2.2.3.1 Subdivision Surface for Modeling and Animation

In 1998, the artistic charm of subdivision surfaces was expressed stunningly for the first time in Geri's Game, which brought the attention of implementing this new technique as the representation of arbitrary topologies [Kerlow 04].

In the spectrum of 3D content enhancement [Junkins and Hux 00], catering to the requirements of preserving more features of models, subdivision surface technology is utilized for constructing meshes with higher resolution by subdividing a coarse mesh in an off-line process. Once the refined mesh is constructed, system resources used for performing subdivision are released. Then the refined mesh with high details can be applied in some real-time application on higher performance processors.

In order to achieve frame-to-frame coherence in visual quality of images' rendering, adaptive subdivision of static and dynamic meshes is implemented as the adaptive adjustment of 3D contents to economize rendering resources [Junkins and Hux 00]. One related application in computer graphics is to display remote objects using lower resolution meshes, while using high resolution meshes to display nearby objects.

In the domain of video games, subdivision surfaces also play an increasingly important role in rendering and animating complex geometric models. In order to build the first subdivision surface engine for Playstation 2®, Brickhill

proposed a practical implementation method to calculate the vertices' positions in different levels rapidly in Brickhill 2001.

To enable the capability of accelerating the rendering and manipulation of massive data effectively, subdivision surfaces are considered to be implemented in hardware such as GPU (Graphic Processor Unit). Toward hardware implementation of subdivision surface, a novel algorithm, which was proved to require a small and constant amount of memory without depending on the subdivision depth, was exploited in Bischoff et al. 00. To achieve higher performance in interactive graphics with improvement on bandwidth, in 2002, Bolz and Schröder proposed to precompute surface tessellations on the CPU by making use of subdivision surface built through linear combination of basis functions, and then use these tessellations to evaluate the surface mesh at runtime in a simple computation performed entirely on a programmable graphics processor [Bolz and Schröder 02].

2.2.3.2 Subdivision Surface for Engineering Design

Subdivision surface are used in engineering design for geometric modeling, design and complex surface reconstruction [Dubé et al. 05]. The utility of subdivision surfaces particularly for engineering applications is investigated and summarized in Gonsor and Neamtu 01.

2.2.3.3 Subdivision Surface Technology for Data Transmission and Compression

Subdivision surfaces can be used as a strategy called as "fill out" [Junkins

and Hux 00] for compressing 3D complex models for their transmission over the Internet (see Figure 2-20). In this case, a server transmits lower resolution meshes. Once smart clients receive them, subdivision surface techniques are used to refine coarse meshes into finer high resolutions meshes. This strategy can reduce the burden of data transmission on the Internet. A dynamic client-server algorithm for selective refinement in the context of network communication including local networks and geographic networks is proposed in Floriani et al. 00, which proves that MT (Multi-Triangulation) for triangle mesh manipulation is a useful Web technique of transmitting massive triangle meshes.

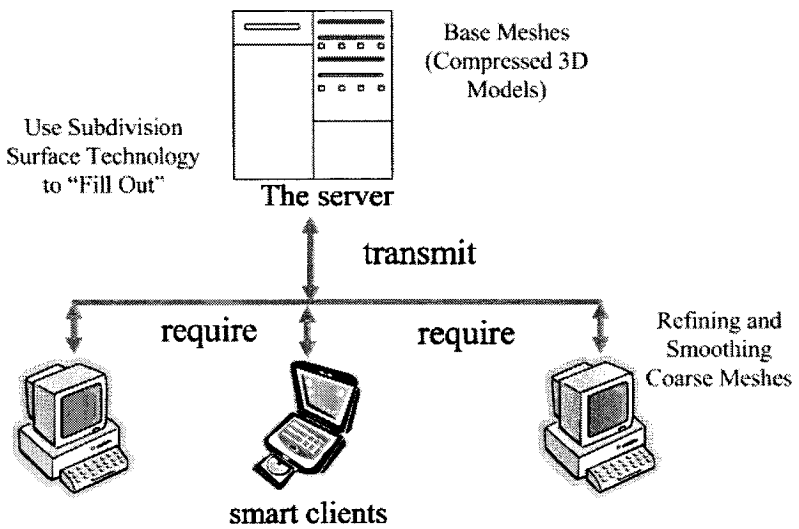


Figure 2-20: Data transmission over the Internet uses subdivision surface technique.

2.2.4 The Motivation of Implementing Subdivision Surfaces

In Pellacini 05, the characteristic of four common surface representations:

implicit surface, parametric surface, polygon meshes and subdivision surfaces are indicated and compared by using table 2-2.

Table 2-2: The general comparison of surface representations [Pellacini 05].

Surface / Characteristic	Implicit Surface	Parametric Surface	Polygon Meshes	Subdivision Surface
Accurate	Yes	Yes	No	Yes
Concise	Yes	Yes	No	Yes
Intuitive Specification	No	Yes	No	No
Local Support	No	Yes	Yes	Yes
Affine Invariant	Yes	Yes	Yes	Yes
Arbitrary Topology	No	No	Yes	Yes
Guaranteed Continuity	Yes	Yes	No	Yes
Parameterization	No	Yes	No	No
Efficient Display	No	Yes	Yes	Yes
Efficient Intersection	Yes	No	No	No

From above discussions, it can be concluded that subdivision surfaces come from high-order smooth surface primitives. With the reference to table 2-2, surface meshes specify large sections of surface using only a few control points, allowing the preservation of smoothness, in contrast with polygon meshes. Moreover, subdivision surfaces supply an attractive practical tool for modeling arbitrary topology, contrary to parametric surfaces. So, in some ways, subdivision surfaces can be viewed as the connection and unification of parametric patches and polygon meshes.

Even though from table 2-2, it sounds like comparing with parametric surfaces, subdivision surfaces lack advantages in parameterization and intuitive specification. However, strictly speaking, parameterization is not part of subdivision surfaces. And if we seek to ameliorate the interactive editing capability of subdivision surfaces, intuitive specification can be improved. Thus, to enable this conceptive improvement come true, the motivation of our project can be initialized as trying to extend the interactive editing capability of subdivision surfaces in multi-resolution category.

Furthermore, the features of subdivision surfaces address many important computational and topological issues that computer graphics practitioners confront. The motivation of implementing subdivision surfaces in our project also comes from the consideration of their features, which can be summarized as follows [Zorin et al. 97, Zorin et al. 00]:

Simplicity & Efficiency: The idea of recursive refinement in sequence makes the acquisition and implementation of subdivision surfaces easy and simple. Because of the characteristic linear property of the subdivision method, the execution of subdivision schemes is efficient.

Arbitrary Topology & Uniformity of Representation: Subdivision solves the issue of surface patches representations with arbitrary topologies, which supplies a practical management for dealing with the lack of continuity across the boundaries between patches. Subdivision extends the affiliation

between polygonal meshes and surface patches, and provides a flexible representation at multiple levels of resolutions.

Scalability: The linear structure of subdivision naturally accommodates multi-resolution algorithms, and data structures, which implies that subdivision surfaces effectively support the manipulation and rendering with different levels of detail.

Numerical Stability: The nice properties such as local support and affine invariance, supply a favorable tool to solve design problems in engineering and computer graphics.

2.3 Work Related to Multi-resolution & LOD

According to the above discussions of subdivision surfaces, multi-resolution technology, which provides an intuitive and a comprehensive implementation of subdivision surfaces' underlying features, is naturally introduced into our project. Multi-resolution surface meshes supply a structural framework for generating, manipulating and rendering meaningful representations of complex geometric models at various levels of detail. The multi-resolution and "Level of Detail" (LOD) technologies are useful resources for real-time graphics applications especially in improving medical visualization and simulation engines, accelerating rendering systems for interactive editing and increasing authenticity of virtual reality systems for entertainment and

education.

In order to provide an introduction to multi-resolution technology in more details, we put the relevant methods into two sections: multi-resolution modeling and LOD.

2.3.1 Multi-resolution Modeling

Multi-resolution relates to the use of variable mesh resolutions [Armin et al. 02, Floriani and Magillo 02]. Resolution represents the density of mesh vertices and the accuracy of polygon meshes. Multi-resolution meshes provide multiple alternative mesh-based approximations describing, at the topmost levels, essential properties and characteristic of a spatial object, and at the most refined levels, fine details and features.

The basic concepts of multi-resolution modeling begin with the introduction of hierarchical geometric models for visible surface algorithms [Clark 76]. This introduction represents an attempt to utilize hierarchical models in a general structural framework for providing various amounts of details in a scene with coherence frames.

Works related to multi-resolution modeling (see Figure 2-21) can be classified into two categories: surface simplification and surface refinement.

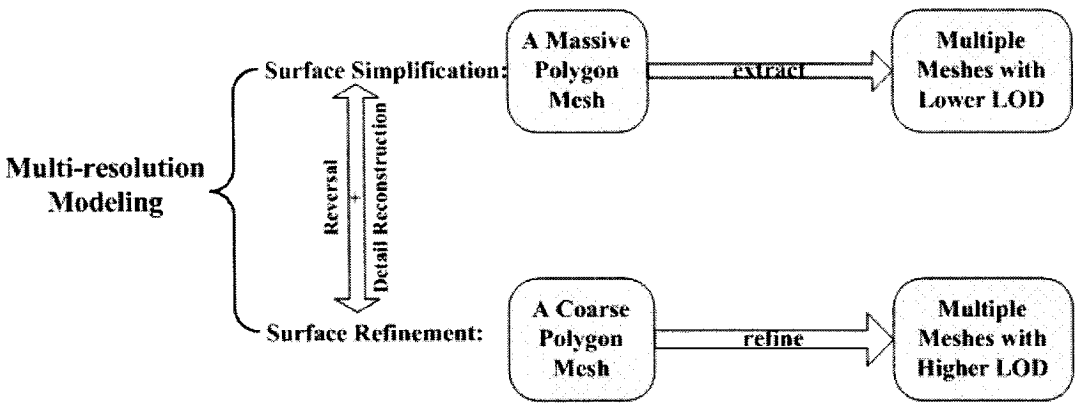


Figure 2-21: Multi-resolution modeling classified into two categories.

2.3.1.1 Surface Simplification

Surface simplification is to take a massive polygonal model as an original mesh and extract multiple approximate simplified models with lesser details by implementing simplifying algorithms.

The works related to simplifying algorithms of polygon meshes have been given a brief introduction in section 2.1.4. In 1997, Herkbert and Garland systematized a variety of methods [Heckbert and Garland 97] for simplifying and approximating polygonal surfaces from cartography, GIS (Geographic Information System), virtual reality, computer vision, computer graphics, scientific visualization, CAGD (Computer Aided Geometric Design), computational geometry and other fields. Their taxonomy is classified according to primary *problem characteristics* and *algorithm characteristics*. *Problem characteristics* are listed as: topology and geometry of input, other attributes of input (ex. color, texture and surface normal), domain of output vertices, structure of output triangulation, approximating elements, error

metric and constraints on solution. *Algorithm characteristics* are Speed/Quality Trade-off and Refinement/Decimation.

In Gotsman et al. 02, recent approaches for compact representations of 3D meshes are surveyed. Simplification and compression methods are summarized according to dissimilarity in: reducing the complexity of polygonal meshes, resampling the meshes' geometry with the optimization of vertex distribution, representing compactly the mesh connectivity and the mesh geometry. In Alliez and Gotsman 04, the survey on recent advances of 3D mesh compression covers further technologies on single-rate and progressive mesh compression.

2.3.1.2 Surface Refinement

Surface refinement is to take a coarse polygonal model as an original mesh and then refine this mesh into a denser model with more LOD by implementing subdivision algorithms.

Works related to subdivision schemes have already been classified in section 2.2. In Warren and Weimer 02 and Zorin et al. 00, the distinct approaches of subdivision-based multi-resolution modeling are described at length. The practical implementation of subdivision-based multi-resolution techniques supplies a good example in this area [Brickhill 01].

2.3.2 Level of Details

Level of Details relates to the accuracy of a mesh to approximate a geometric model [Armin et al. 02, Floriani and Magillo 02]. The common form of LOD is used in rendering distant objects by simplifying the amount of detail [Luebke et al. 03]. The history of LOD follows the development of multi-resolution modeling technology all along.

In Luebke et al. 03 and Luebke et al. 02, historical genesis, earlier applications and current approaches of LOD are overviewed in details. Meanwhile, various run-time LOD frameworks for managing level of detail such as selecting (optimization-based predictive schedulers), discrete (traditional decoupling simplification and rendering), continuous (progressive meshes) and view-dependent LOD (vertex hierarchies), are introduced comprehensively. In addition, algorithms and operators for generating models by simplification, error metrics for evaluating LOD creation, run-time management on selection criteria, popping prevention, frame rates and resources (ex. memory, internet streaming) are discussed both in theory and in practice.

2.3.3 A Brief Summary of Recent Approaches

In multi-resolution modeling, the researches are approached in multi-resolution analysis, control of LOD, algorithms and data structure in decimation and refinement of meshes.

In 1999, Garland surveyed and commented the future of multi-resolution modeling [Garland 99 - Eurographics 99] with the structural introduction and comparison of Image Pyramids, Volume Methods, Vertex Decimation, Vertex Clustering, Edge Contraction, Simplification Envelopes and Wavelet Surfaces. Simultaneously, Schröder emphasized the opportunities of subdivision-based multi-resolution modeling [Schröder 99] referring to the techniques of remeshing, irregular subdivision and combined subdivision schemes.

In 2002, Floriani and Magillo presented models and data structures for multi-resolution meshes [Floriani and Magillo 02, Armin et al. 02] based on spatial objects, which implies the continuous improvements and gradual maturity of multi-resolution technology. Later, Adjacency and Incidence Framework (AIF) for dynamic multi-resolution meshing algorithms [Silva and Gomes 03] was proposed as an efficient data structure to manage multi-resolution meshes. In Shaffer and Garland 05, a powerful data structure with an efficient access to multi-resolution meshes and the flexible operation on massive geometric data was built.

In 2003, the publication of *Level of Detail for 3D Graphics* [Luebke et al. 02] from the Morgan Kaufmann Series in Computer Graphics and Geometric Modeling, supplied comprehensive references in the evaluation and optimization of multi-resolution meshes, which predicates the fine incorporation and vigorous future between LOD technology and multi-resolution technology.

In the following years, as the interactive editing of multi-resolution meshes improved, remeshing technique [Alliez et al. 05], with a comprehensive coverage of the issues: validity, quality, fidelity, discrete input, large data sets, uncertainty and correspondence, and with the consideration of algorithmic requirements in LOD, complexity and theoretical guarantees, in this field also flourished.

2.4 Work Related to Interactive Mesh Editing

Using polygonal meshes to represent surfaces is ubiquitous in geometric modeling. In order to visualize meshes with level of detail, multi-resolution meshes are generated to manage quality surface approximations in multi-levels. Besides, due to the requirements for manipulating surface meshes, interactive mesh editing methods challenge the tremendous qualifications on remeshing and rendering massive mesh data flexibly and efficiently. Thus, the researches on multi-resolution hierarchies for interactive mesh editing are brought to the forefront of 3D geometric modeling.

In the spectrum of multi-resolution modeling, around 1997, a multi-resolution representation for subdivision-based meshes, is first proposed for a scalable interactive multi-resolution editing system [Zorin et

al. 97], combining subdivision and smoothing algorithms. The analysis, synthesis, and rendering algorithms with adaptive and local operators, to edit arbitrary polygonal meshes are implemented in order to obtain adequate interactivity. It allows large scale mesh editing with the definite adjustment of details. Multi-resolution transforms including the corresponding important properties such as smoothness, locality and simplicity are also discussed. This approach is at the cost of extra time to analyze and extra memory to preserve its hierarchical structure.

In 1998, the concept of multi-resolution modeling is generalized to arbitrary triangle meshes by using incremental mesh decimation schemes without subdivision connectivity [Kobbelt et al. 98]. The umbrella algorithm combining local frame coding and multi-level smoothing, allows modifying meshes interactively. However, this approach in Kobbelt et al. 98 limits mesh modification to operations such as rotation, scaling and translation. In Suzuki et al. 98, an adaptive remeshing method is proposed to evaluate the deformation of surfaces caused by dragging parts of a 3D triangular mesh. However, the important issue about multi-resolutions meshes has not been covered in Suzuki et al. 98.

In 1999, Lee presents a new approach to multi-resolution mesh editing. The editing process is based on harmonic maps (mapping editing areas into 2D rectangles) and multi-level B-Spline interpolation (updating the modifications in 2D back to 3D meshes) [Lee 99]. However, instead of manipulating control points of multi-resolution meshes directly; this method only handles mapped vertices in 2D rectangles.

In 2000, in comparison with WEDS (Winged-Edge Data Structure) and PEL (Point-to-an-Edge-List), Madi and Walton propose TLDS (Triangular-Loop Data Structure) to represent geometric models, providing cross-reference information of polygon meshes' vertices [Madi and Walton 00]. Besides, an algorithm to transform hierarchical representations of geometric models into TLDS representations is described in this paper. This approach allows immediate access to neighbouring polygons of selected vertices, which results in efficient shape deformations.

In 2001, generating sharp features on multi-resolution subdivision-based surfaces [Biermann et al. 01] by tagging previously defined curves is implemented in mesh editing operations such as trimming. In Praun et al. 01, an algorithm for the consistent parameterization established as an element of DGP (Digital Geometry Processing), is proposed to resample and remesh multiple geometric models with different shapes. As a result, this algorithm allows building the same connectivity and sampling pattern onto various models.

In 2002, algorithms for intuitive cut-and-paste editing of multi-resolution surfaces at interactive rates are explored [Biermann et al. 02], which enhance the robustness of manipulation tools. The limitations of this approach lies in the moderate correspondence between the identified target regions and chosen source surfaces.

In 2003, *displacement volumes* [Botsch and Kobbelt 03] as a new detail

surface representation for multi-resolution models is proposed, by utilizing volume elements enclosed between successive surfaces at different levels. This approach is proved to be efficient and robust to reconstruct deformed surfaces with adequate preservation of natural details.

In 2004, Adjacency and Incidence Framework (AIF) is implemented to edit multi-resolution meshes by using sub-meshes on a logical sequence of steps: Mesh Simplification, Sub-mesh Creation and Editing, Mesh Refinement [Silva and Gomes 04]. The novel idea of this approach is to specify confined sub-meshes upon subdivision.

In 2005, a new progressive multi-resolution representation for deforming surface meshes allowing dynamic incremental improvements to the hierarchy with high quality surface approximations at any LOD via reclustering is presented [Kircher and Garland 05]. This innovative approach facilitates the multi-resolution technology with effective updated operations for the area of animation.

In summary, a large number of researches address the issue of interactive editing of multi-resolution meshes. They can be outlined by the difference of editing operations, deformation operations, data structures or algorithms. Because this issue covers a variety of extensive topics, we only list the representative works according to the order of chronicle.

2.5 Restatement of Our Research

In this chapter, an overview of related works covering large-scale research fields has been presented. Our research is not only inspired by these advanced issues, but also further extended from these representative researches.

In our project, we implement two subdivision schemes: the Loop subdivision and the Modified Butterfly subdivision as the basis to generate subdivision-based surface meshes at multi-levels. Considering the Loop scheme as the representative of the approximating schemes and the Modified Butterfly scheme as the representative of the interpolating schemes, we analyze their differences in details.

With reference to the former researches on boundaries/creases, we improve the masks used for boundaries/creases by verifying surface meshes with different topologies. Moreover, we expand the crease application in two operations: vertex crease and edge crease.

We implement a triangle Quadtree similar to the one discussed in Zorin et al. 00 as the basic data structure in our project. The innovation of our work is to evaluate its performance by implementing two subdivision schemes and to summarize time and memory costs by carrying out specific subdivision tasks.

To enhance intuitive specification of subdivision surfaces mentioned in section 2.2.4, the algorithm used for editing subdivision-based surface meshes is ameliorated. Different from the related works discussed in section 2.4, the novelty of our work is to keep all the changes from different subdivision levels when performing subdivision operators, not only from the lower levels (parent inheritance), but also from the higher levels (children retrospect), while still considering the effective local support within the multi-resolution framework with LOD queries.

With the aim of establishing a friendly GUI (Graphics User Interface), our project explores the interface for displaying and manipulating surface meshes and possible functions used in the interactive editing. The contribution of our research is to extend the possible methods for manipulating surface meshes interactively.

CHAPTER 3

IMPLEMENTATION OF MULTI-RESOLUTION SURFACE MESH EDITING

In this chapter, the techniques for implementing two types of subdivision surfaces, approximation surface meshes generated from the Loop scheme and interpolation surface meshes generated from the Modified Butterfly scheme, are presented. In the framework of multi-resolution technology, the interactive manipulation of subdivision-based surface meshes is illustrated through generation and editing of multi-resolution surfaces.

In section 3.1, the Loop and the Modified Butterfly schemes are detailed, and the relevant differences between approximating subdivision surface and interpolating subdivision surface are summarized. In section 3.2, crucial methods to manipulate subdivision surface meshes such as defining creases to keep sharp features, and keeping changes from any subdivision level particularly from higher levels while implementing local support, are addressed. In section 3.3, data structure used to implement subdivision schemes and the algorithms used to edit multi-resolution subdivision surfaces are presented.

3.1 Subdivision Schemes

In this section, the Loop and the Modified Butterfly schemes are introduced. In our project, we implement these two stationary face-split subdivision schemes. Face-split subdivision can be described as subdividing triangle meshes by splitting each triangle facet into four. Both subdivision schemes involve inserting new vertices into each edge of triangular facets and connecting the new vertices at a given subdivision level, where these additional new vertices are referred as *odd vertices* and those retained old vertices are referred as *even vertices* [Zorin et al. 00]. Thus performing these two subdivision schemes can be described as two processes: (1) Subdividing the mesh to generate new faces and insert new vertices, which updates mesh topology; (2) applying corresponding rules to calculate and place relevant vertices on surface meshes, which updates mesh geometry.

The loop scheme is an approximating scheme, which generates surfaces with C^2-continuity over regular parts except at extraordinary vertices, where the limit surface remains C^1-continuous. Differently, the Modified Butterfly scheme is an interpolating scheme, which produces surfaces with arbitrary topology achieving C^1-continuity in most cases [Zorin 98]. The different subdivision rules of these two schemes to compute or modify surface vertices produce significant differences in the appearance of surfaces. Generally speaking, the Loop subdivision surface meshes possess nicer smoothness, whereas the Modified Butterfly subdivision surface meshes possess less

pleasing smoothness.

In the next two sections, the detailed rules of the two subdivision schemes are introduced. And then an observation of surface quality obtained from the implementation of these two schemes is summarized in section 3.1.3.

Before we go into particulars, we give a brief explanation of the notations and terms for subsequent use.

Valence. Sometimes the valence of a vertex can be referenced as "vertex degree", which marks the number of edges derived from the appointed vertex. The new vertices generated during subdivision have the same valences, which are known as *regular* vertices. Otherwise, vertices of any other valences are known as *extraordinary* vertices. For triangle subdivision schemes, interior vertices with valence 6 and vertices on the boundaries with valence 4 are regular. For most cases, extraordinary vertices are not generated during subdivision, but are inherited from the original control meshes.

Masks. Specific subdivision rules are applied to calculate new vertex positions in consideration of the old vertices on the former surface mesh. The influence of old vertices to new vertices is defined by masks consisting of coefficients or eigenvalues.

Boundary and Crease. Generally, a limit surface can be constructed from interior vertices and vertices on the boundaries. Contrary to interior vertices,

specific rules are defined to calculate vertices on the boundaries. Typically, any interior control vertex is not supposed to influence the boundaries on the same limit surface. As the extension of surface mesh's applications, sometimes interior edges can be tagged as creases and the special rules for vertices on boundaries are applied for vertices on creases.

3.1.1 Loop Scheme

The Loop scheme leads to smooth surfaces by recursively applying approximating subdivision rules on triangle meshes. Loop subdivision defines specific rules for different parts of triangle meshes: interior, boundaries and creases.

Interior. On the Loop subdivision surfaces, each interior odd vertex is influenced by the two extreme vertices of the selected edge and another two neighbouring vertices which define the two neighbouring triangles sharing the selected edge. And each interior even vertex is influenced by its neighbouring vertices and its former position.

Figure 3-1 (a) and Figure 3-1 (b) illustrate respectively the masks for interior odd vertices and interior even vertices. The updating process of the odd vertices and the even vertices on the subdivision triangle meshes of a cone model is demonstrated in Figure 3-2.

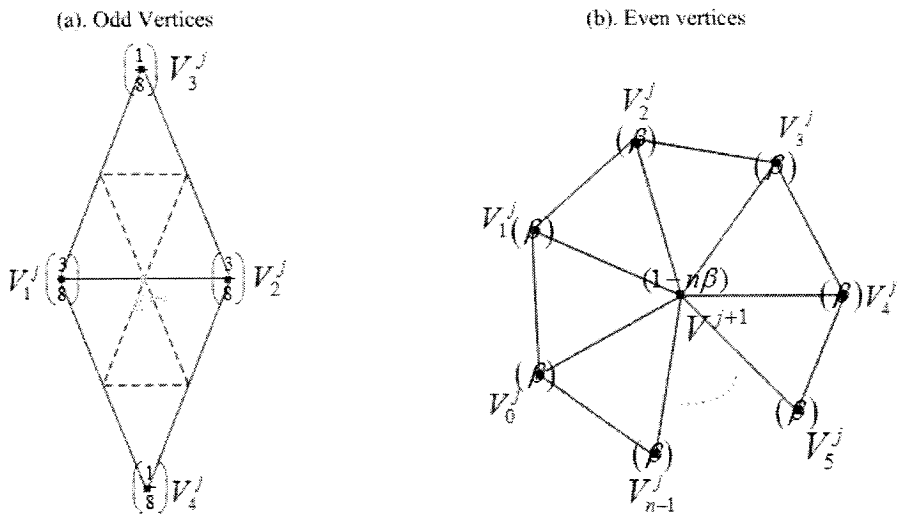


Figure 3-1: The masks for interior odd vertices and interior even vertices of the Loop subdivision scheme [Zorin et al. 00].

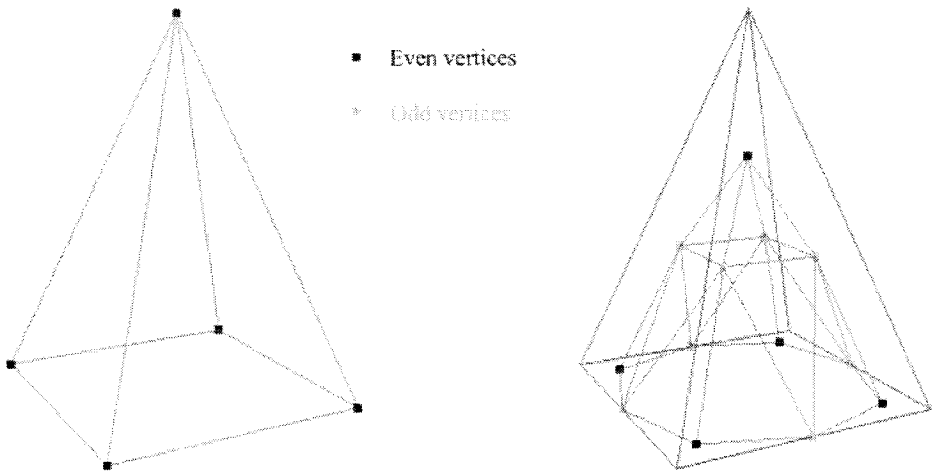


Figure 3-2: The updating process of odd vertices and even vertices on Loop subdivision triangle meshes of a cone model.

The rule for computing interior odd vertices (E^{j+1}) is described in Equation 3.1, where V_i^j and E_i^j represent i^{th} vertex at j^{th} subdivision level and n represents

vertex valence (where $i, j, n \in N$).

$$E^{j+1} = \frac{3}{8}(V_1^j + V_2^j) + \frac{1}{8}(V_3^j + V_4^j); \tag{3.1}$$

Similarly, the rule for computing interior even vertices (V^{j+1}) uses Equation 3.2, where the coefficient β is involved in the calculation of the masks' eigenvalues, and is used to define weights of even vertex's neighbouring vertices (V_i^j) and its former vertex (V^j).

$$V^{j+1} = (1 - n\beta)V^j + \beta \sum_{i=0}^{n-1} V_i^j; \tag{3.2}$$

The definition of β influences the continuity and smoothness of surface meshes. The formula presented as Equation 3.3 was originally proposed in Loop 87.

$$\beta = \frac{1}{n}\left(\frac{5}{8} - \left(\frac{3}{8} + \frac{1}{4}\cos\frac{2\pi}{n}\right)^2\right); \tag{3.3}$$

However, Loop's subdivision scheme is proved to have one problem: discontinuous but bounded curvatures emerge on surface meshes, when valence of vertices n is 4 and 5 [Warren and Schaefer 04]. In Joe Warren's monograph on subdivision [Warren 94], a simple corrected version of β was proposed to solve the problem mentioned above [Warren and Weimer 02]. In

our project, we implement β based on Warren's version (see Equation 3.4).

$$\beta=\begin{cases}\frac{3}{16}, & n=3;\\ \frac{3}{8n}, & n>3;\end{cases} \tag{3.4}$$

Boundaries. The masks for odd vertices and even vertices on boundaries, which were first proposed in Hoppe et al. 90, are illustrated in Figure 3-3.

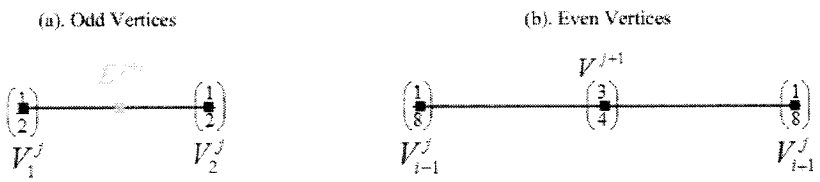


Figure 3-3: The masks for odd vertices and even vertices on boundaries [Zorin et al. 00].

Each new odd vertex E^{j+1} on boundaries is defined only by the two extreme vertices of the selected edge (see Figure 3-3(a)). The calculation mask of odd vertices are weighted as $(0,\frac{1}{2},\frac{1}{2},0)$ instead of $(\frac{1}{8},\frac{3}{8},\frac{3}{8},\frac{1}{8})$. The rule for computing odd vertices on boundaries uses Equation 3.5:

$$E^{j+1}=\frac{1}{2}V_1^j+\frac{1}{2}V_2^j; \tag{3.5}$$

In the same way, each even vertex is only influenced by the vertices on the selected edges, and is mainly influenced by its former position (see Figure

3-3(b)). The rule for computing even vertices on boundaries uses Equation 3.6:

$$V^{j+1} = \frac{1}{8}V_{i-1}^{j} + \frac{3}{4}V_{i}^{j} + \frac{1}{8}V_{i+1}^{j}; \tag{3.6}$$

Creases. In order to keep sharp features of specific parts on surface meshes, some interior edges can be tagged as creases. Then creases are dealt with as boundaries by reducing the influence from the neighbouring vertices during subdivision. In our project, we define creases on the Loop subdivision surface meshes to edit surfaces. Considering different influence from the neighbours, we generate two types of crease: vertex crease and edge crease. The concrete implementations about defining creases are presented in the section 3.2.

3.1.2 Modified Butterfly Scheme

In 1990, the Butterfly scheme was first introduced in Dyn et al. 90. The name of "Butterfly" comes from the configuration of the eight-point mask or stencil used for its calculation. But this original version only interpolated edges' endpoints linearly, which results in undesirable jagged cracks at extraordinary vertices, whose valences are not six. Later in 1996, Zorin et al. proposed the Modified Butterfly scheme, which adapted the coefficients of subdivision masks to the valence k of one given vertex and took the cases of neighbouring edges and vertices on the boundaries into account [Zorin et al. 96]. The results of implementing the Modified Butterfly scheme are proved to

guarantee C^1-continuity for arbitrary surface meshes even at extraordinary vertices [Zorin et al. 00, Zorin 98].

The Modified Butterfly scheme is based on a similar process as the Loop's scheme to achieve subdivision surfaces at multi-levels by applying subdivision rules recursively on triangle meshes. But the Modified Butterfly scheme is an interpolating algorithm, which does not apply changes to even vertices (old retained vertices) (see Figure 3-4).

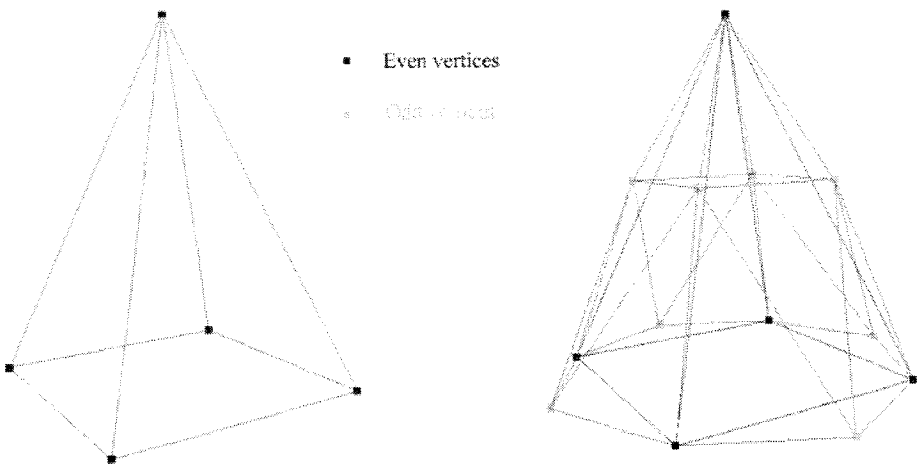


Figure 3-4: Modified Butterfly subdivision triangle cone meshes with odd/even vertices.

In contrast with the Loop scheme, the masks in the Modified Butterfly scheme used for calculating odd vertices are much more complicated, which can be distinguished as: *(a)* for interior odd vertices with regular neighbours, *(b)* for odd vertices with regular neighbours on interior or boundary/crease, and *(c)* for odd vertices adjacent to an extraordinary vertex or with neighbours of an extraordinary vertex on boundary/crease [Zorin et al. 98]. Moreover, the case *(b)* includes at least 4 types of particular stencils for

different cases. And the case *(c)* includes 2 different cases qualified by two corresponding complex stencils which impact on the results of the scheme's implementation as a whole.

In our project, we implemented the Modified Butterfly scheme according to Zorin et al. 00 in the beginning; however, it did not achieve the ideal results. As our implementation evolved, we found that some of the coefficients in the subdivision masks of Zorin et al. 00 had not been correctly addressed. Later, we found that one improved version was proposed in Attene et al. 05, which presented the handle of recovering curved sharp edges in triangle meshes. Finally in our implementation, we updated the improved version with small changes according to both references and verified them using many different models especially the ones whose neighbouring vertices and edges of odd vertices are on the boundaries. The final results are proved to achieve certain fineness on surface meshes.

With reference to Zorin et al. 00 and Attene et al. 05, we present all the rules in three categories of cases. In each category, first we present the general masks which adapt to most cases, and then we append the introduction of the other special cases.

Case *(a)*:

The standard Butterfly mask for interior odd vertices with interior regular neighbours (see Figure 3-5):

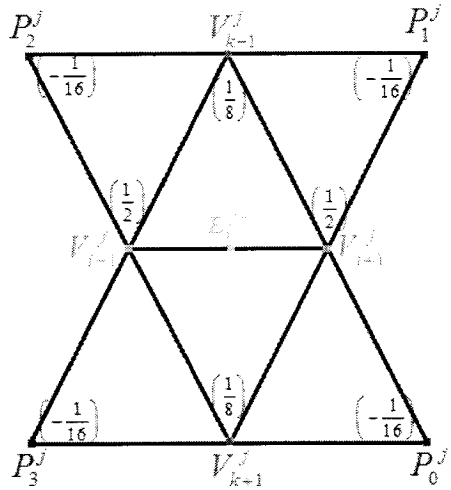


Figure 3-5: Standard Butterfly stencil [Zorin et al. 00].

And its rule can be found in Equation 3.7:

$$E_i^{j+1} = \frac{1}{2}(V_{i-1}^j + V_{i+1}^j) + \frac{1}{8}(V_{k-1}^j + V_{k+1}^j) - \frac{1}{16}(P_0^j + P_1^j + P_2^j + P_3^j); \tag{3.7}$$

Case *(b)*:

(1). The mask for odd vertices on boundary or crease with regular neighbours (see Figure 3-6):

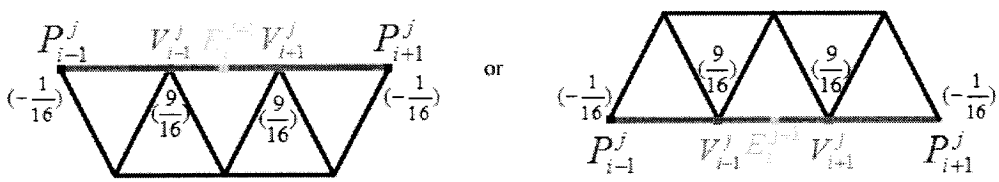


Figure 3-6: The mask (the 4-point stencil) for odd vertices on boundaries or creases with regular neighbours [Zorin et al. 00].

And its rule can be found in Equation 3.8:

$$E_i^{j+1} = \frac{9}{16}(V_{i-1}^j + V_{i+1}^j) - \frac{1}{16}(P_{i-1}^j + P_{i+1}^j); \tag{3.8}$$

(2). With the exception of the 4-point stencil, the special masks for neighbours with regular valences on interior or boundaries/creases include 4 special rules, but the non-manifold crease rule is seldom used (see Figure 3-7). In our project, we do not consider the non-manifold crease rule and we only list it here with reference to Attene et al. 05.

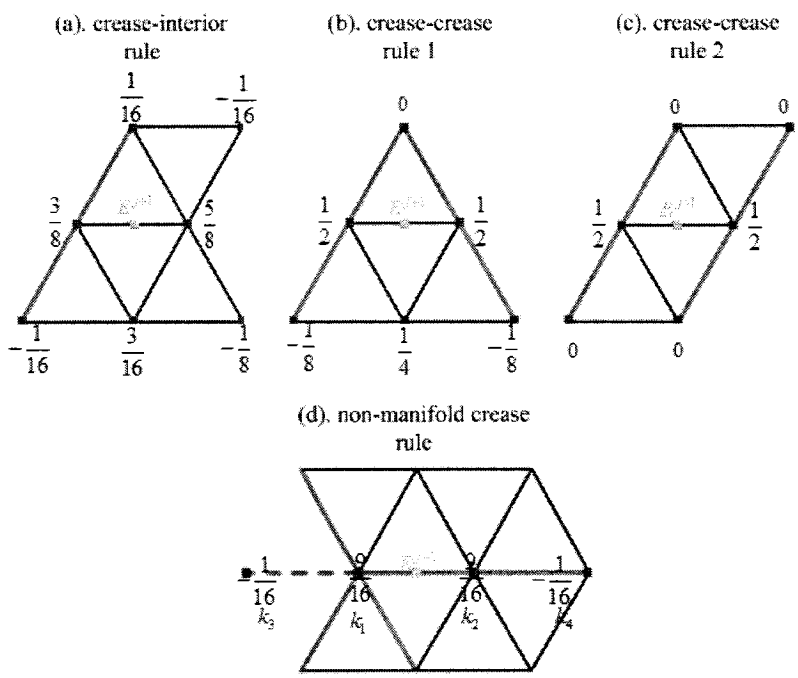


Figure 3-7: Four special masks for neighbours with regular valences on interior or boundaries [Zorin et al. 00 & Attene et al. 05].

For the non-manifold crease rule, k_1, k_2, k_3, k_4 illustrated in Figure 3-7(d)

presents each selected vertex valence. Their relation satisfies Equation 3.9. When both two vertices where k_1 and k_2 represent are manifold sharp vertices, Equation 3.8 is used.

$$k_3 = k_4 + (k_1 - k_2)\bullet(\frac{2(k_1 - k_2)\bullet(k_2 - k_4)}{(k_1 - k_2)\bullet(k_1 - k_2)} + 1) \tag{3.9}$$

Case *(c)*:

(1). The mask for odd vertices adjacent to an extraordinary vertex (see Figure 3-8), which represents two cases: one neighbouring vertex is a regular interior and the other neighbouring vertex is an extraordinary interior, or one neighbouring vertex is an extraordinary vertex on boundary/crease and the other neighbouring vertex is a regular vertex on boundary/crease.

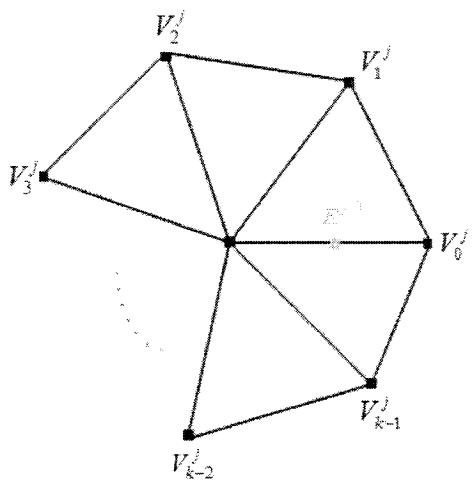


Figure 3-8: Extraordinary-interior rule [Zorin et al. 00].

In Figure 3-8, V_i^j and E^{j+1} represent vertices on surface meshes. Their

corresponding weights can be found in Equation 3.10:

$$
\begin{aligned}
&k=3: \\
&E^{j+1}=\frac{3}{4}E^{j}+\frac{5}{12}V_0^{j}-\frac{1}{12}(V_1^{j}+V_2^{j}); \\
&k=4: \\
&E^{j+1}=\frac{3}{4}E^{j}+\frac{3}{8}V_0^{j}-\frac{1}{8}V_2^{j}; \\
&k\geq 5: \\
&E^{j+1}=\frac{3}{4}E^{j}+\sum_{i=0}^{k-1}(\frac{1}{k}\bullet(\frac{1}{4}+\cos\frac{2\pi i}{k}+\frac{1}{2}\cos\frac{4\pi i}{k})\bullet V_i^{j});
\end{aligned}
\tag{3.10}
$$

(2). The mask for extraordinary-crease rule is illustrated in Figure 3-9, which represents the cases where one neighbouring vertex of the new odd vertex is a regular interior and the other neighbouring vertex is an extraordinary vertex on boundary/crease, or one neighbouring vertex is a regular vertex on boundary/crease and the other one is an extraordinary vertex on boundary/crease.

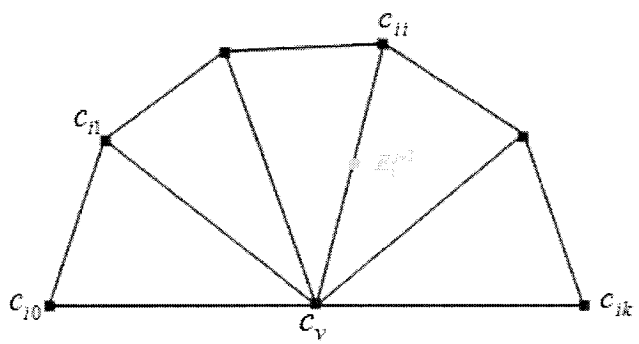


Figure 3-9: Extraordinary-crease rule [Zorin et al. 00 & Attene et al. 05].

In Figure 3-9, c_{ij} represents the j^{th} coefficient of the i^{th} vertex and E^{j+1}

represents the new odd vertex at $(j+1)^{th}$ level. By verifying whether the sum of all the relevant coefficients in the mask is one, we modified the corresponding mask in our project as Equation 3.11:

$$
\begin{aligned}
& k = valence(\textit{the vertex with the coefficient } c_v); \\
& 0 \le i, j \le k; \\
& \theta = \frac{\pi}{k-1}; \\
& c_v = 1 - \frac{(\sin(\theta) \bullet \sin(i\theta)}{k \bullet (1-\cos(\theta))}; \\
& c_{i0} = -c_{ik} = \frac{1}{4}\cos(i\theta) - \frac{\sin(2\theta)\sin(2i\theta)}{4k(\cos(\theta)-\cos(2\theta))}; \\
& c_{ij} = \frac{1}{k} \bullet ((\sin(i\theta)\sin(j\theta) + \frac{1}{2} \bullet \sin(2i\theta)\sin(2j\theta));
\end{aligned}
\tag{3.11}
$$

3.1.3 The Comparison between Two Subdivision Schemes

After the implementation of the Loop subdivision and the Modified Butterfly subdivision, we summarize some differences between them.

1. From above, we can conclude that the rules to calculate odd vertices in the Modified Butterfly subdivision are much more complicated. But the Loop subdivision involves more calculation for even vertices. We use the same examples to test the time costs of executing the two schemes. And we receive nearly the same costs for the two absolutely different types of computing (see Table 4-3 and Table 4-4, Table 4-6, Table I-7). After

researching and analyzing, we conclude that even though the Modified Butterfly costs much more time for calculations of masks, compared with the time used for memory management, structure filling and connectivity's calculation, this cost is negligible.

2. Because for the Loop scheme, the old vertices are non-stationary, its subdivision surfaces at higher levels do not pass through the control points generated at lower levels. Thus, sharp features are much more obvious when we create creases on Loop subdivision-generated surfaces. While for the Modified Butterfly scheme, the old vertices remain stationary. Considering the basic control points inherited from original meshes, the shapes of the Modified Butterfly subdivision surfaces are mainly retained. That's why it is less obvious that using the Modified Butterfly subdivision reshapes surface meshes.

3. Comparing with the fineness of the Loop subdivision surface meshes, the smoothness of the Modified Butterfly scheme is less satisfying. It is much more difficult to verify the surfaces' smoothness and the validity of the scheme's implementation, especially when the masks used for extraordinary vertices on the Modified Butterfly subdivision surfaces are so varied.

4. The limit Loop subdivision-based surfaces look like a "shrinking" of its original mesh that converges to a limit surface entirely inside the bounding box of the original mesh. While the limit Modified Butterfly subdivision-based surface resembles an "inflation" of its original mesh

(see Figure 3-10). That's why the convergence pace for the limit Loop subdivision mesh is much more rapid and the shapes of the final Loop subdivision meshes are much smaller than their original meshes, while the shapes of the final Modified Butterfly subdivision meshes are a little bigger than their original meshes (see Figure IV-1 and Figure IV-2 in Appendix IV).

5. When the original surface meshes are denser and close with fewer boundaries, implementing the Modified Butterfly subdivision can achieve nicer fineness. Thus, to achieve surfaces smoothness, the implementation of the Modified Butterfly scheme depend more on the density and the topology of the original surface meshes (see Figure 3-11). We have found that an excellent combination is to first perform Loop subdivision on the original surfaces to get denser subdivision surfaces, and then to perform Modified Butterfly subdivision. Consequently, two types of subdivision schemes do not act against each other; on the contrary, they can cooperate to generate better results (see Figure 3-12, Figure 3-13).

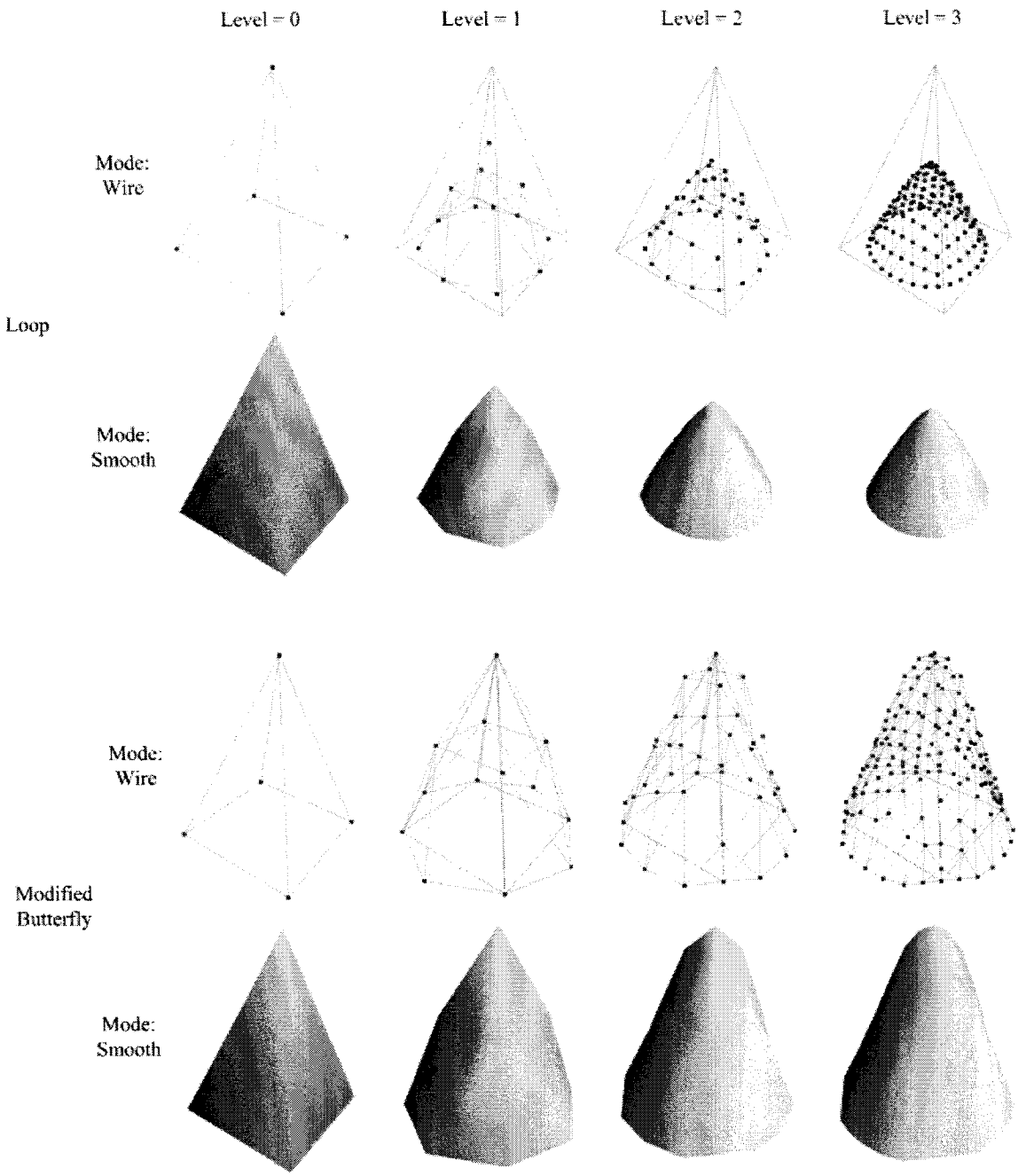


Figure 3-10: Visual comparison between performing Loop subdivision and performing Modified Butterfly subdivision on the same initial cone mesh.

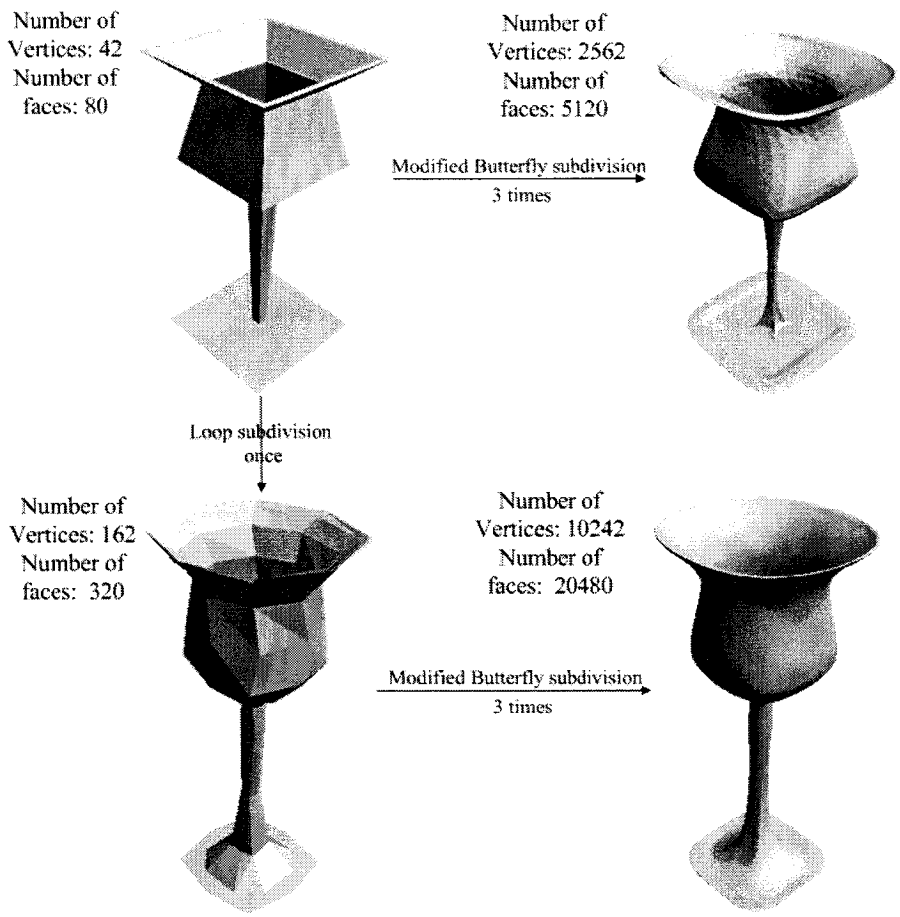


Figure 3-11: The comparison of performing the Modified Butterfly subdivision on the sparse surface and on the denser surface.

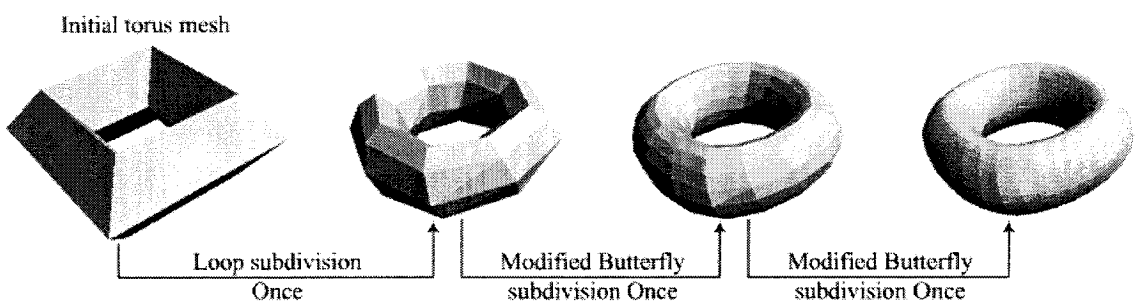


Figure 3-12: The illustration of cooperating two schemes on the torus mesh.

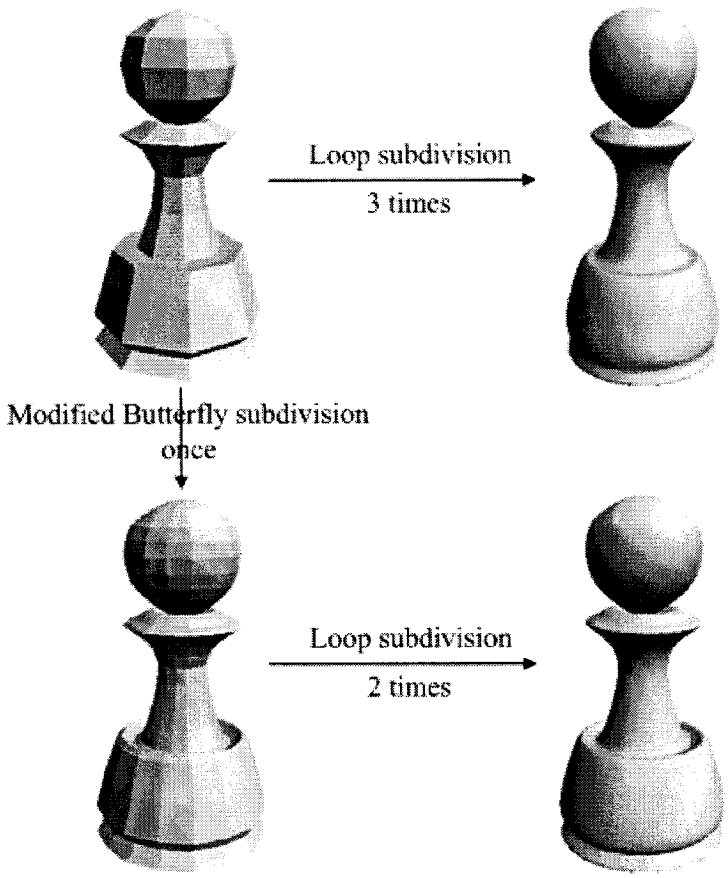


Figure 3-13: Visual comparison between only performing Loop subdivision and cooperating two schemes on the same initial pawn mesh.

3.2 The Implementation of Editing Subdivision Surfaces

In the multi-resolution subdivision framework, we implement different methods to edit subdivision-based surface meshes, such as generate creases and modify pre-selected control points in our project. To render and edit

surface meshes conveniently, an environment with a windows GUI (Graphic User Interface) was built. Considering the properties of the subdivision schemes, this GUI meets the requirement of displaying subdivision meshes at multi-levels while modifying them and keeping the subdivision or editing results.

3.2.1 Multi-resolution Subdivision Framework

Multi-resolution subdivision can be viewed as firstly, recursively subdividing an original mesh, which generates multiple subdivided surface meshes at different levels; secondly, adjust LOD (Level-Of-Details) in different levels (see Figure 3-14).

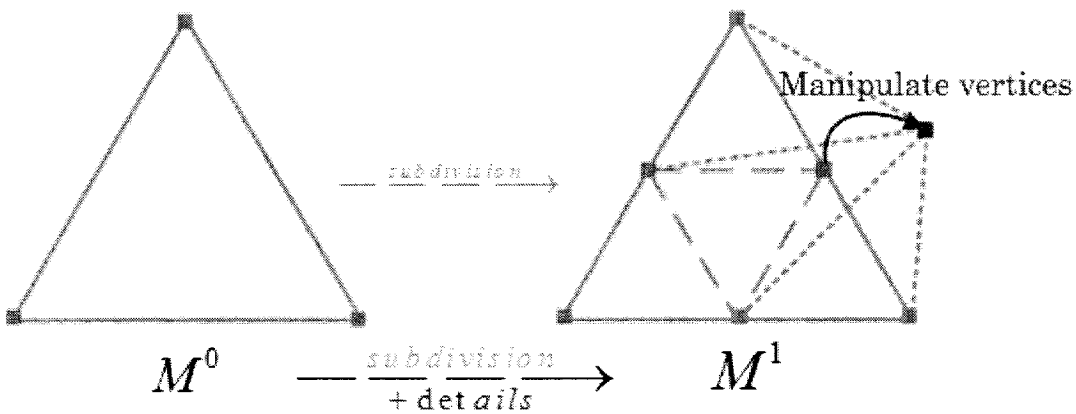


Figure 3-14: The sequence of implementing multi-resolution subdivision.

Accordingly this sequence can be expressed as Equation 3.12, where M^j represents the multi-resolution surface mesh, S^j represents the surface

mesh and $(\Delta d)^j$ represents the difference adjusted at the corresponding j^{th} level; $f(\,)$ represents a subdivision operator, which varies according to the subdivision scheme implemented:

$$M^j = f(S^j) + (\Delta d)^j; \quad (j \in N) \tag{3.12}$$

The validity of Equation 3.12 can be deduced from the essential property of the subdivision operator: affine invariance, which we mentioned in the section 2.2.1. Evidently, the subdivision implied in surface's editing supplies us with a framework to efficiently manipulate subdivision-based surface meshes at multi-levels (see Figure 3-15).

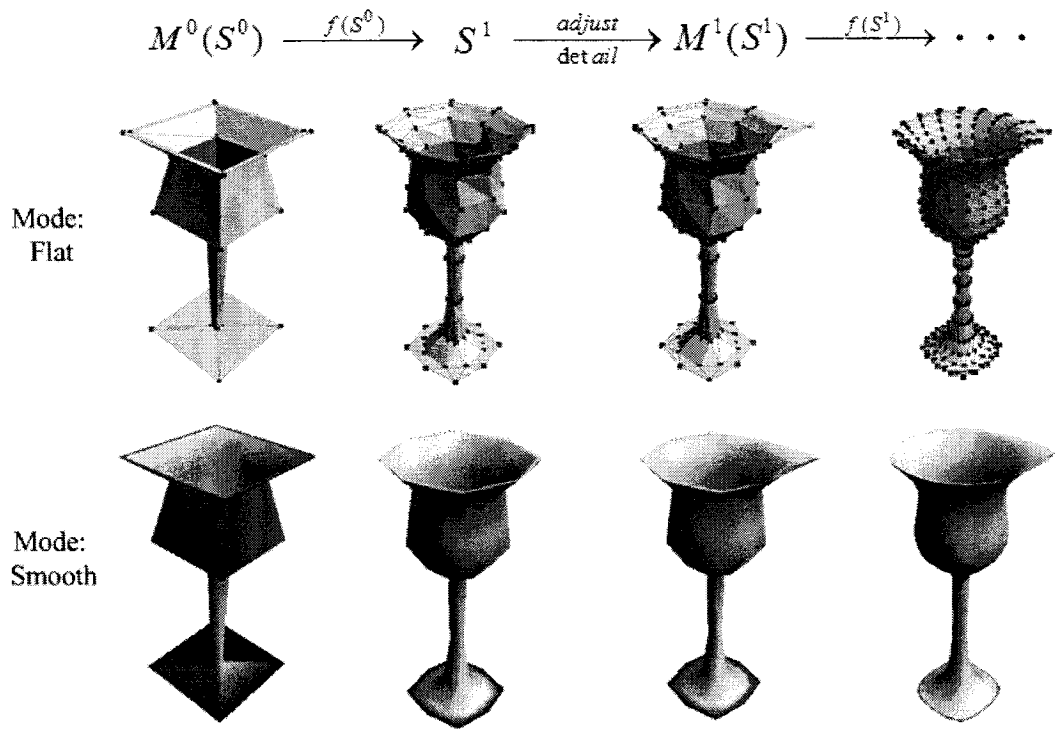


Figure 3-15: The cup surface meshes based on Loop subdivision are edited at different subdivision levels.

3.2.2 Manipulate Sharp Features

In our project, the "Create Crease" operation for editing surface is implemented, which manipulates sharp features following a group of pre-defined edges on a multi-resolution surface. In Biermann et al. 01, the novelty of crease's applications within the multi-resolution subdivision framework is explored very well.

In general, creases can be viewed as the boundaries between surface mesh patches. The basic idea of creating creases is to keep sharp features of interior edges, so that even after iterated subdivision, the parts of surface meshes are not going to be smoothed completely. Defining creases is up to real-time users' requirements and specific applications; while in substance, generating creases is determined by adjusting the influence of neighbouring vertices around user-defined edges to keep the sharp features on the surface.

In our project, we implement two types of crease creation: vertex crease and edge crease on the Loop subdivision surface meshes. Meanwhile, we extend edge crease to Modified Butterfly subdivision surfaces.

(1). Vertex crease:
"Vertex crease" is to keep the selected vertices unchanged during subdivision. Thus when the selected vertices remain at the same positions as on previous levels, the influenced neighbouring vertices around these selected vertices

perform the relevant preservation; while at the same time, these neighbours are relocated approximately according to the subdivision masks. One example of defining one vertex is illustrated in Figure 3-16 and another example of defining multiple vertices is illustrated in Figure 3-17.

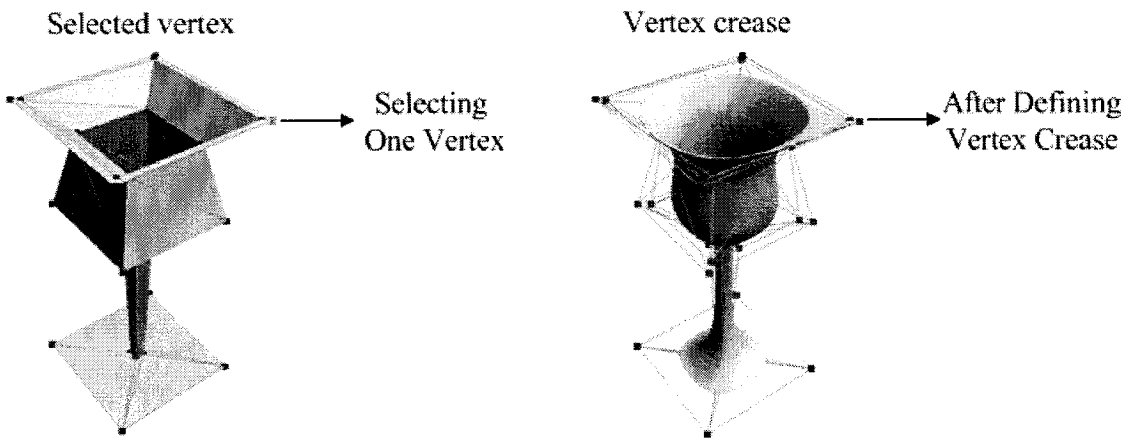


Figure 3-16: Selecting and defining one vertex as vertex crease on the cup surface mesh.

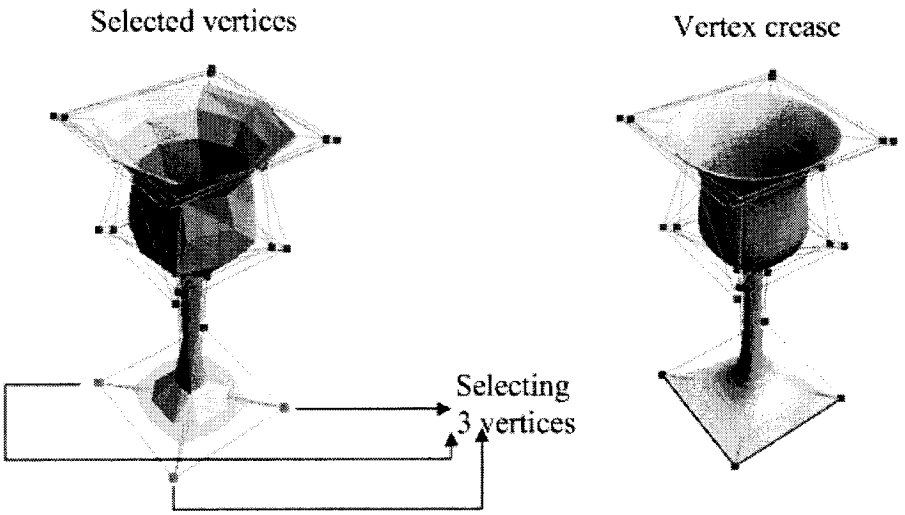


Figure 3-17: Selecting and defining multiple vertices as vertex creases on the cup surface mesh.

(2). Edge crease:

"Edge crease" is to tag the selected edges as boundaries (see Figure 3-18). Thus the selected edges are defined as creases and applied the same masks as boundaries. Normally, the masks for boundaries can be used directly, however, when two edges or more than two edges are selected together, their topology needs to be considered particularly and specials cases need to be specified.

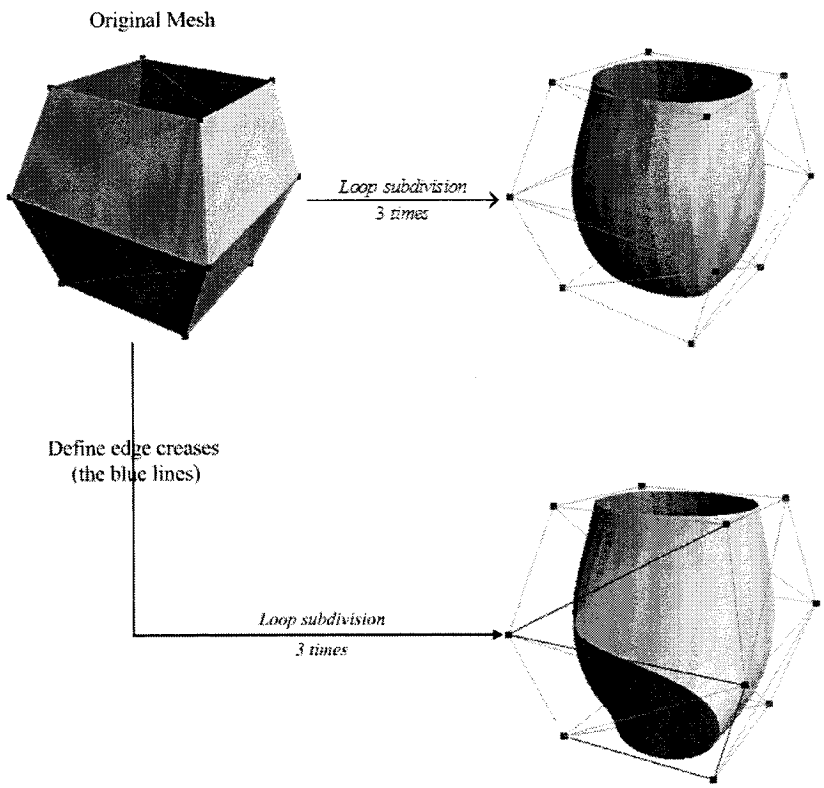


Figure 3-18: Defining edge creases on the vase subdivision surface.

For Loop subdivision, we first observe the specified vertex surrounded by the tagged creases, and then we specify one case as only two sharp edges on each

side of the selected vertex and another one as more than two sharp edges on each side of the selected vertex. In our project, for the first case, we implement regular rules for the even vertices on boundaries. And for the second case, we implement regular rule for interior even vertices but treating each selected edge as a boundary.

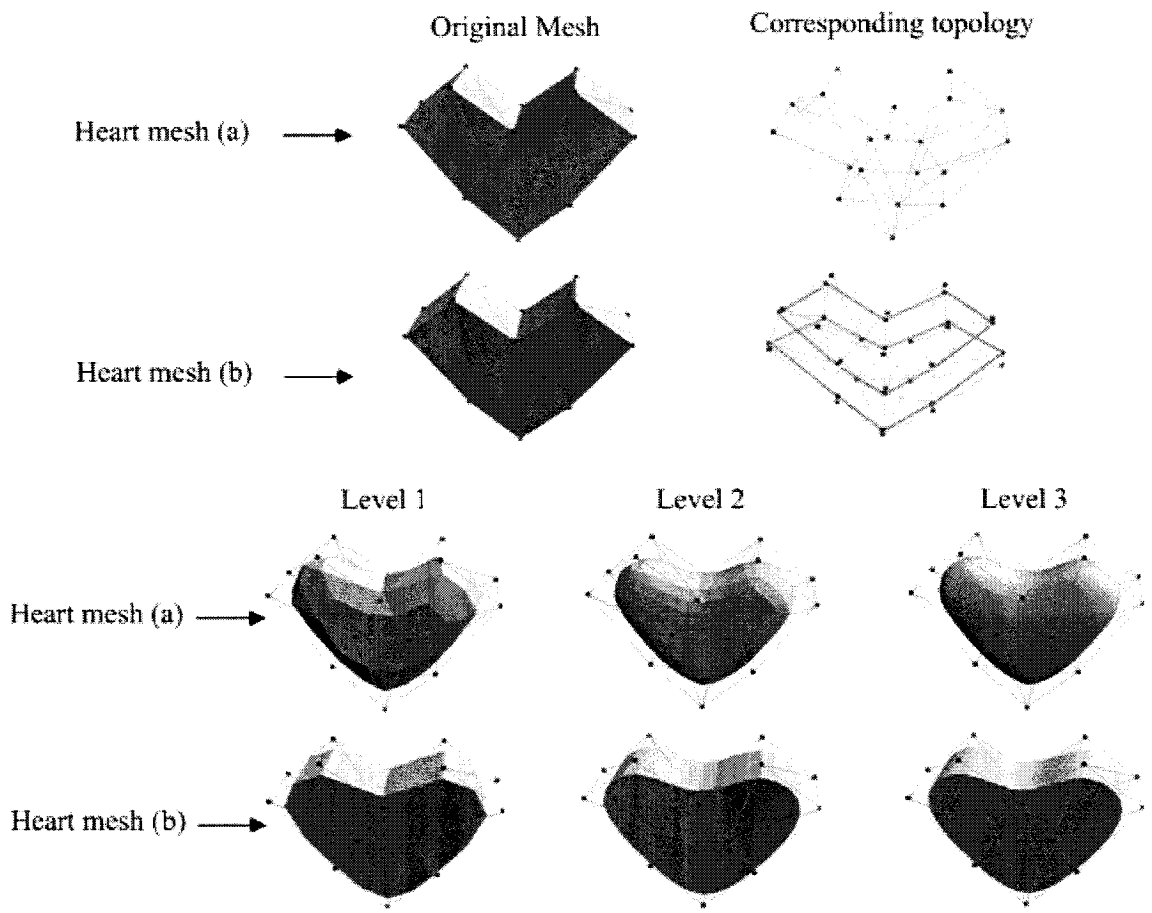


Figure 3-19: Two heart meshes with different topologies.

Generally, the topology of the original mesh influences a lot the subdivision results. Figure 3-19 illustrates the two heart meshes with different topologies, which achieve different results after subdivision. However, sometimes

defining edge creases can bring some special effects. Figure 3-20 illustrates the same example used in Figure 3-19 (the upper row with closed surface meshes – heart mesh (a)) without crease, with vertex crease and with edge crease, where we can find that defining edge creases can achieve the same results as the ones with separate boundaries on the lower row – heart mesh (b) in Figure 3-19.

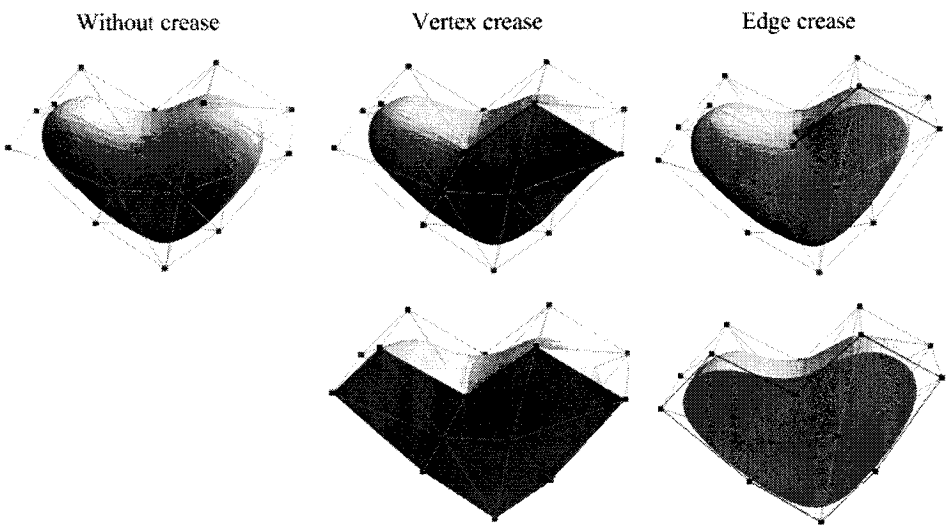


Figure 3-20: Heart meshes without crease, with vertex crease/edge crease.

For Modified Butterfly subdivision, we need to add more specific rules to check whether both vertices are on creases. For different cases, we exactly follow the modified masks for boundaries described in section 3.1.2, and then we achieve nice results when defining edge crease on Modified Butterfly subdivision surface (see Figure 3-21).

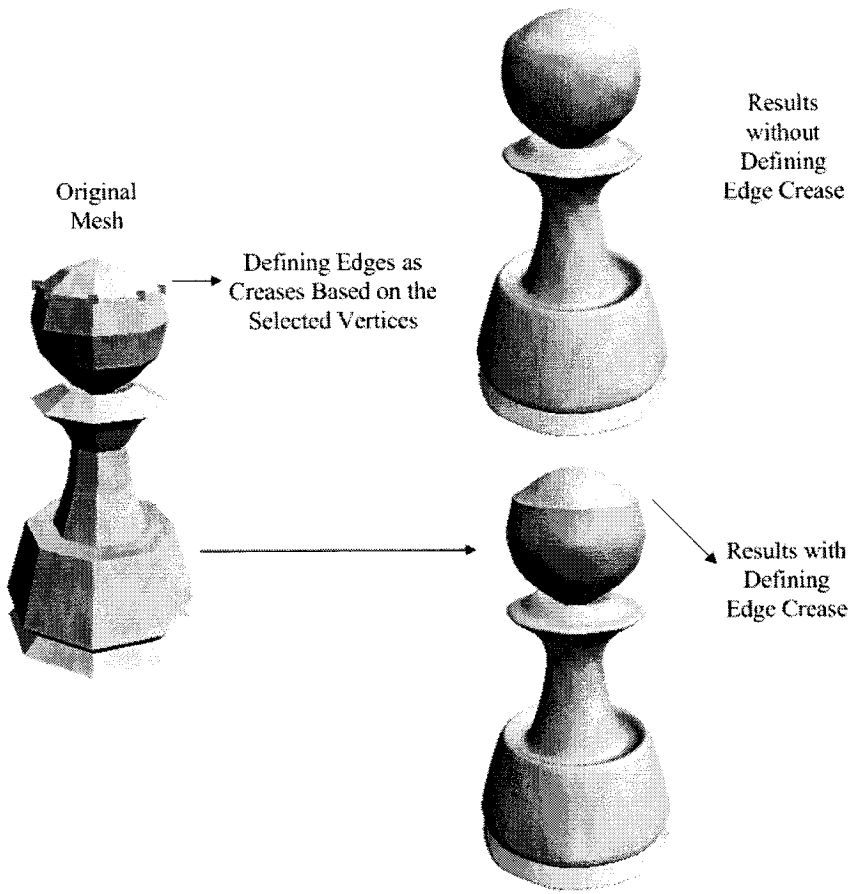


Figure 3-21: Pawn meshes without crease and with edge crease.

3.2.3 Editing Control Points on Multi-resolution Subdivision Surface

When the control points on multi-resolution Loop subdivision surface meshes are being manipulated, it is important to consider two factors: finite or local support and keeping vertex changes at each level. To achieve these two objectives, appropriate data structures and algorithms must be used for the manipulation of subdivision surface meshes at multiple levels. In this part, the methods used for editing and the problem exiting during editing are demonstrated. In section 3.3, data structures and algorithms used to present

methods and solve the problem are going to be discussed.

For interactive manipulation, efficiency plays an important role in the editing process. In fact, when one vertex is edited, the changes only influence its neighbouring vertices. From the observation of the eigenvalue matrix used for calculation of control points at higher levels, we can determine that the influence of neighbouring vertices during the Loop subdivision is local and follows the "one-ring" rule (see Figure 3-22). Namely, an efficient way to update new meshes on higher levels is to only update the influenced neighbouring vertices, instead of recalculating the whole surface mesh.

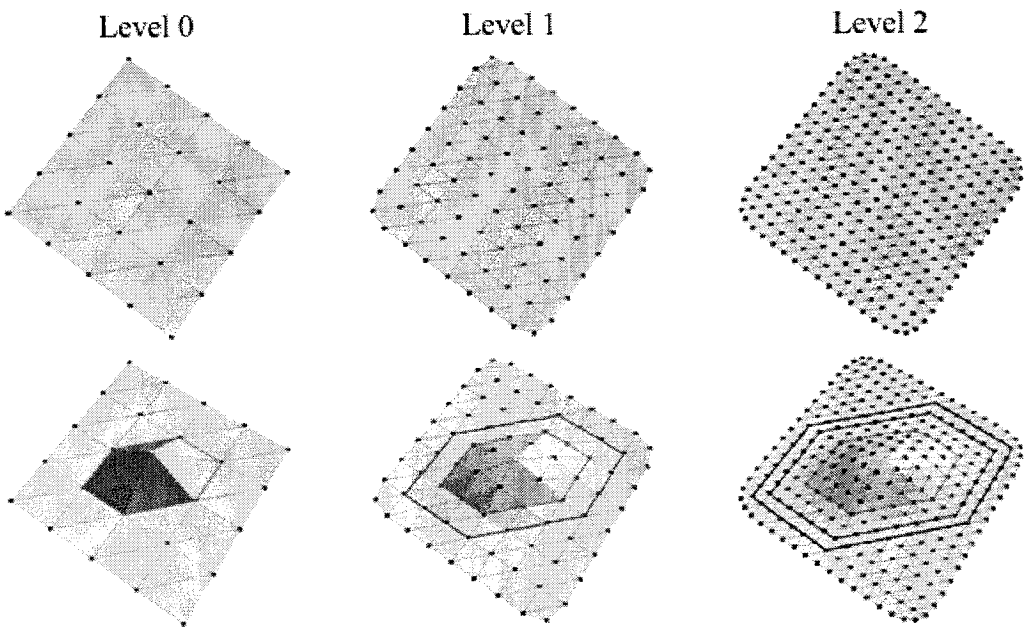


Figure 3-22: Illustration of local support following one-ring rule, where the selected vertices influence the one-ring neighbouring vertices on next level.

Once control points on the current surface meshes, which can viewed as the

parents of vertices on the next subdivided surface meshes, are manipulated, the meshes on higher levels, which can be viewed as children of control meshes on lower levels, are going to be locally regenerated. So it brings forth the question: when the children vertices are updated on higher levels, would the updated information be kept even though the parent vertices are updated again? In our project, we improve the current editing algorithm and supply one solution for the above problem in section 3.3.

3.3 Implementation of Data Structures and Algorithms

In this section, data structures used for implementing the Loop scheme and the Modified Butterfly scheme are briefly discussed. The algorithms used for editing vertices on hierarchical surfaces are also presented.

3.3.1 Data Structures

Data structures used to represent the hierarchical mesh structures generated by subdivision and to manipulate multi-resolution subdivision surface meshes interactively, directly influence the implementation efficiency. A variety of factors such as the effective representation of source data geometry and topology, the efficient update of mesh manipulation and the flexible exploitation of subdivision features determine our selection of data structures.

The global data structure to describe hierarchical meshes in our project, which is similar to the one in Zorin et al. 00, can be presented as a Triangle Quadtree (see Figure 3-23). Considering the properties of subdivision algorithms, the data structure of triangle mesh in our project is constructed with explicit elements: Face and Vertex. While the relative connection information for edges can be traced from the arrays of pointers in Face and Vertex. The data structure for vertex contains the arrays of pointers, which indicates its parent vertex at the coarser level and its children vertices at the finer level. Their basic structure and main relations are presented in Figure 3-24.

Once the vertices are edited, the changes are kept in the additional data structure "Change", which stores the index of the vertex edited in the current mesh and the difference between the former position and the new position. The information reserved is useful to solve the problem which we propose in section 3.2.3 about keeping all the changes, even though the editing of neighbouring vertices is performed from higher levels to lower levels on the multi-resolution surface meshes while still considering the local support.

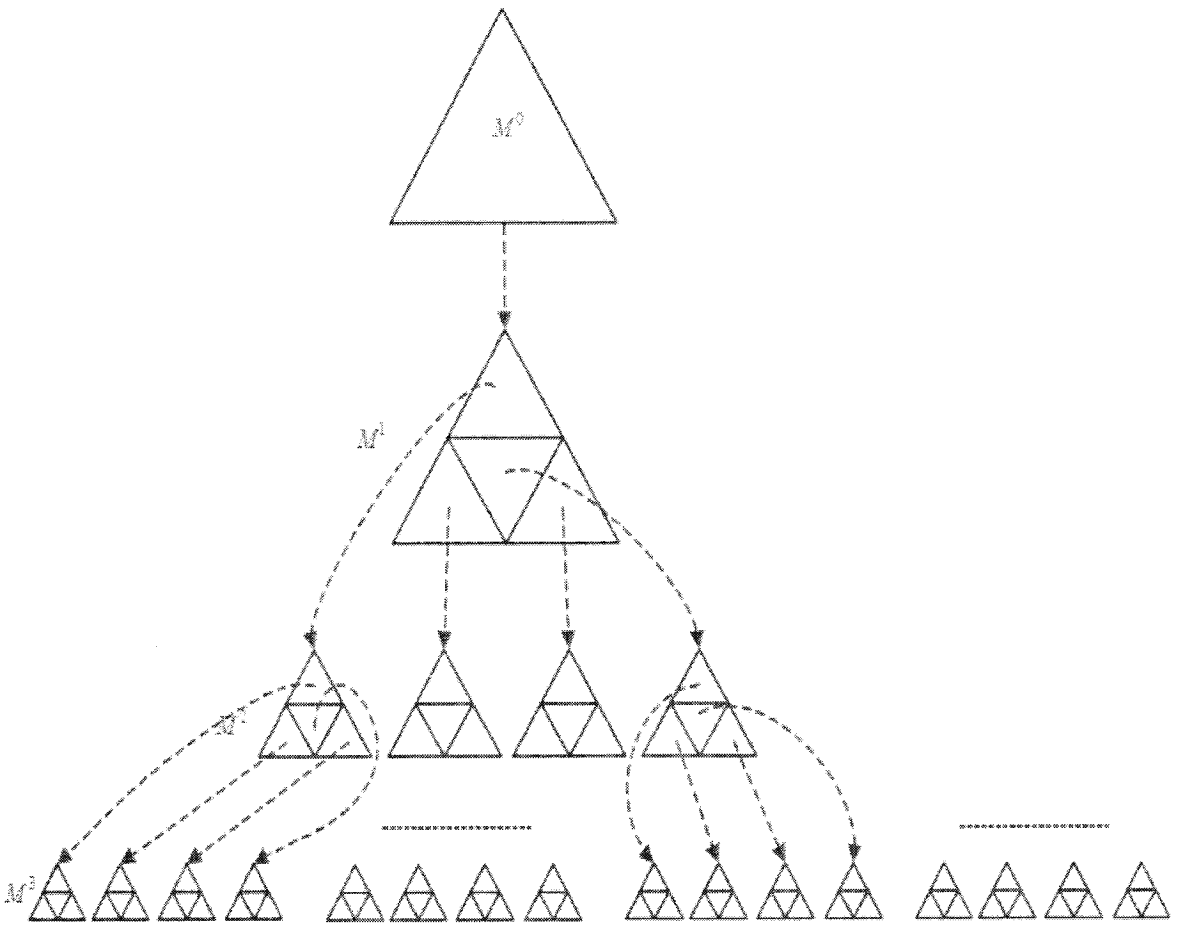


Figure 3-23: Triangle QuadTree as the global data structure for meshes from coarse level to finer levels, where M^i represents the triangle surface at i^{th} level.

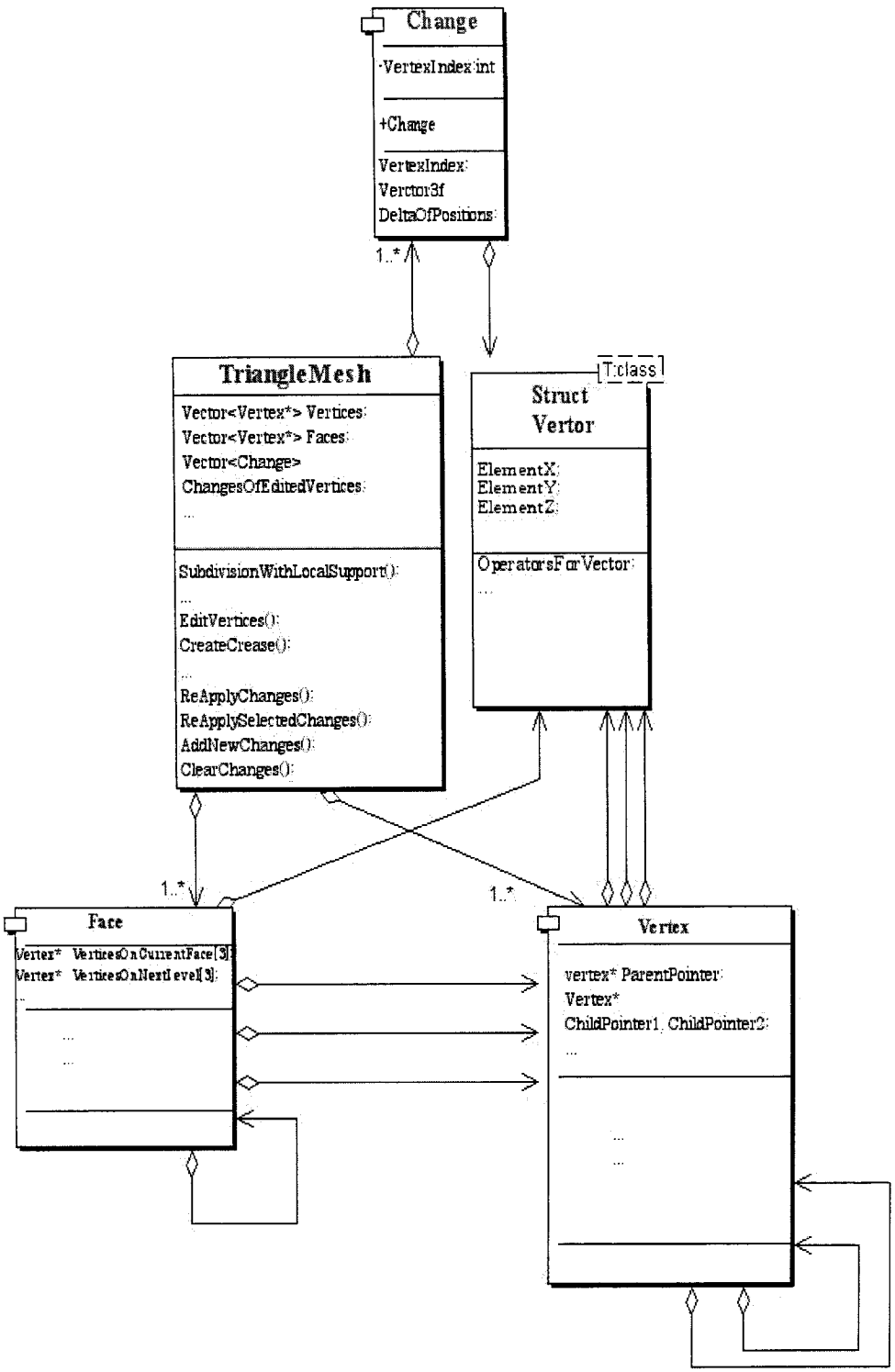


Figure 3-24: The diagram describes the basic structure and the main relations of elementary elements on the Loop subdivision surface meshes.

3.3.2 Algorithms

The procedure for performing multi-resolution surface mesh editing is illustrated in Figure 3-25. In this process, our contribution is to establish the algorithms which preserve the changes that occurred during the editing, while following local support. Our method is to record all the changes while the vertices are being edited, where we call the function *AddNewChanges* (see Figure 3-26). Then when Loop subdivision with local support is performed, we call the function *ReApplySelectedChanges* (see Figure 3-27).

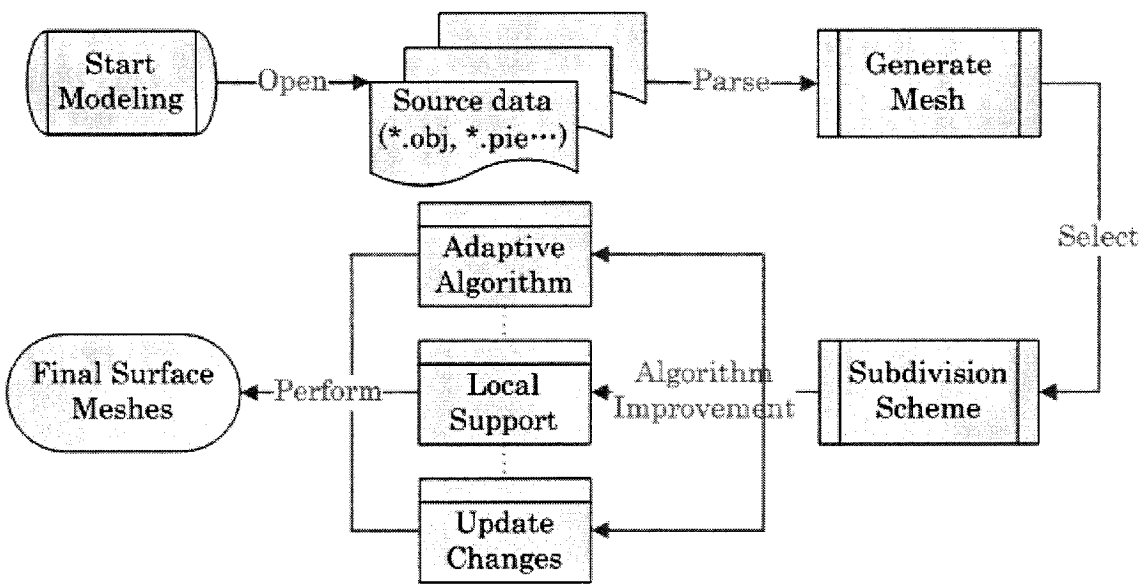


Figure 3-25: The procedure of performing multi-resolution surface mesh editing.

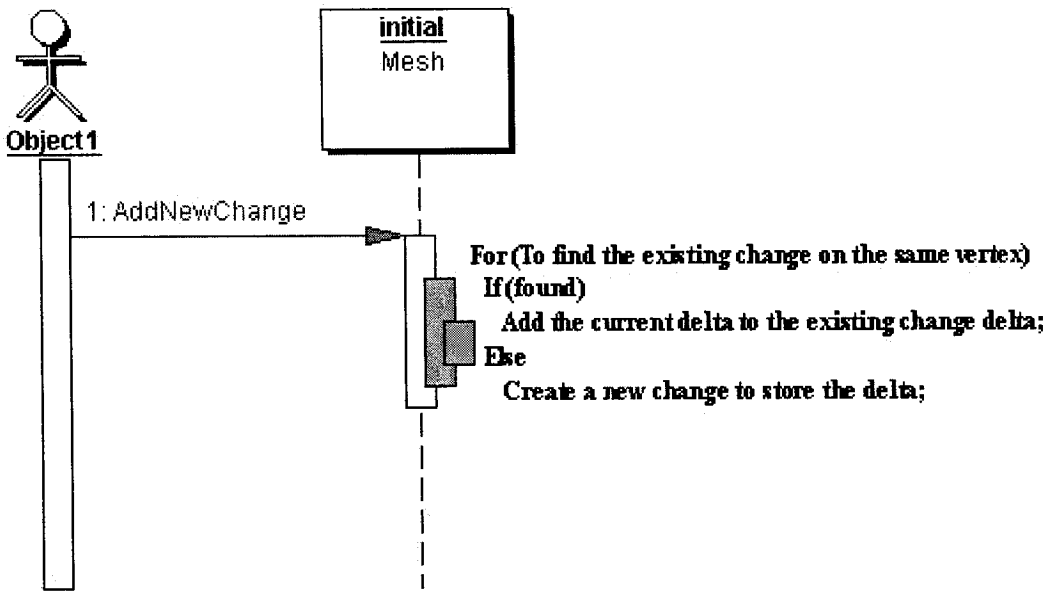


Figure 3-26: The illustration of function *AddNewChange( )*.

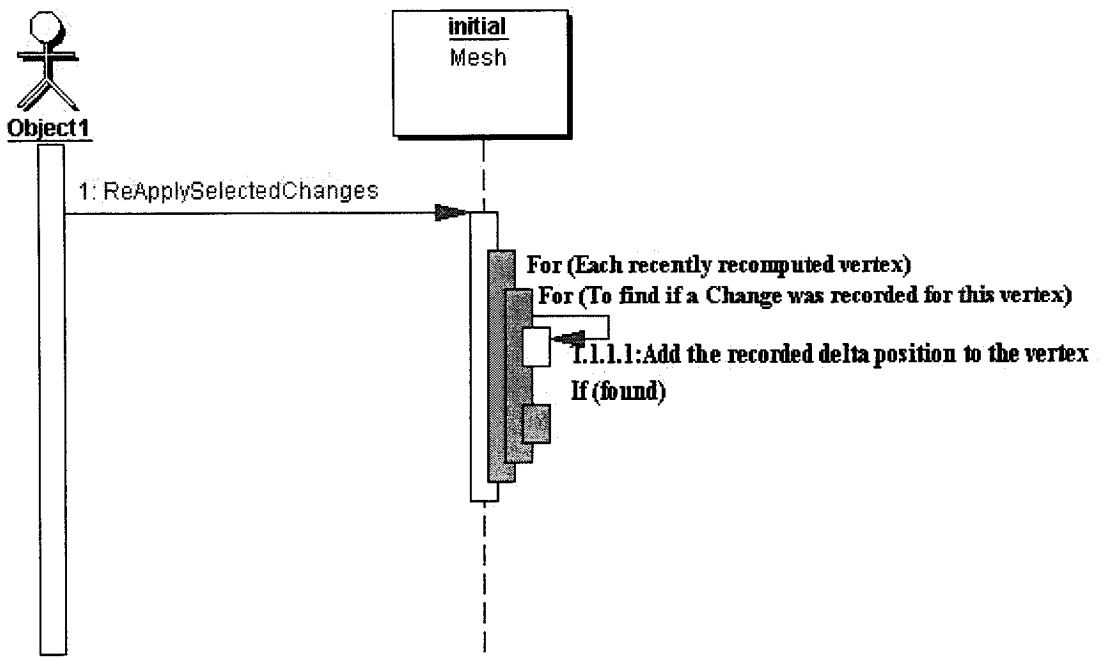


Figure 3-27: The illustration of function *ReApplySelectedChanges( )*.

CHAPTER 4

EVALUATION AND RESULTS

In this chapter, we address system performance evaluation issues within our multi-resolution subdivision surface mesh editing framework. We also elaborate on relevant implementation results.

According to discussions from the previous chapters, we can summarize surface subdivision as an efficient technique to iteratively produce finer limit surface meshes from coarse arbitrary meshes. While this subdivision process is being carried out towards limit surfaces, efficiently handling the subdivision operator brings forth two issues. One issue relates to algorithm and data structure used for effectively implementing subdivision schemes; the other issue relates to performance of operating certain editing functions. Thus, our system performance is evaluated not only by assessing the algorithms' validity and efficiency, but also by analyzing practical processes whereby using the editing system manipulates subdivision surfaces.

Our evaluation begins with the analysis of time costs for calculating mesh connectivity and surface normals, and execution costs to perform corresponding subdivision schemes with different level of details. Then we test the influence of local support, while implementing Loop subdivision. Next, we perform representative interactive manipulations on surface meshes to prove the validity of keeping changes from higher level to lower

level with local support. At the end, we demonstrate the results obtained while attempting to create creases on arbitrary surface meshes.

The results are obtained by performing the relevant evaluations on a PC with a Pentium® 4 CPU 3.20GHz, L2 Cache 2MB, RAM 1GB and NVIDIA GeForce FX 5900 Ultra, which allows displaying certain surface meshes at interactive rate while executing our editing system in real-time.

In the following sections, we first choose some representative control meshes as test examples. Then we perform various procedures to get specific outcomes. Finally, we analyze them and provide comparisons and conclusions.

4.1 Evaluation in the Multi-resolution and Subdivision-based Framework

Different levels of details of surface meshes influence time costs of rendering and editing control meshes. Once a surface mesh is initialized, lists of vertices and lists of faces with connectivity information are stored in memory. Each time a subdivision operation is performed, the relevant connectivity information is updated. So, time cost for computing mesh connectivity supplies a useful reference for performance evaluation. In order to render the smooth appearance of hierarchical surfaces, the surface normals must be

computed properly to shade the meshes. Thus, we provide statistics on time costs for calculating surface normals at each subdivision level. Time cost for performing subdivision is also a very important factor to consider, which determines the global system performance.

At first, a cup surface mesh is used as a test example, whose original control mesh contains regular vertices and extraordinary vertices and whose density (or complexity) is moderate (see Figure 4-1(a)). In this case, hierarchical surface meshes will not bring a huge burden on our editing system. Therefore, iteratively performing mesh generation and interactive editing will not cause obvious system degradation. Moreover, the topology of the cup control mesh can satisfy particular test requirements, including the presence of extraordinary vertices and boundaries.

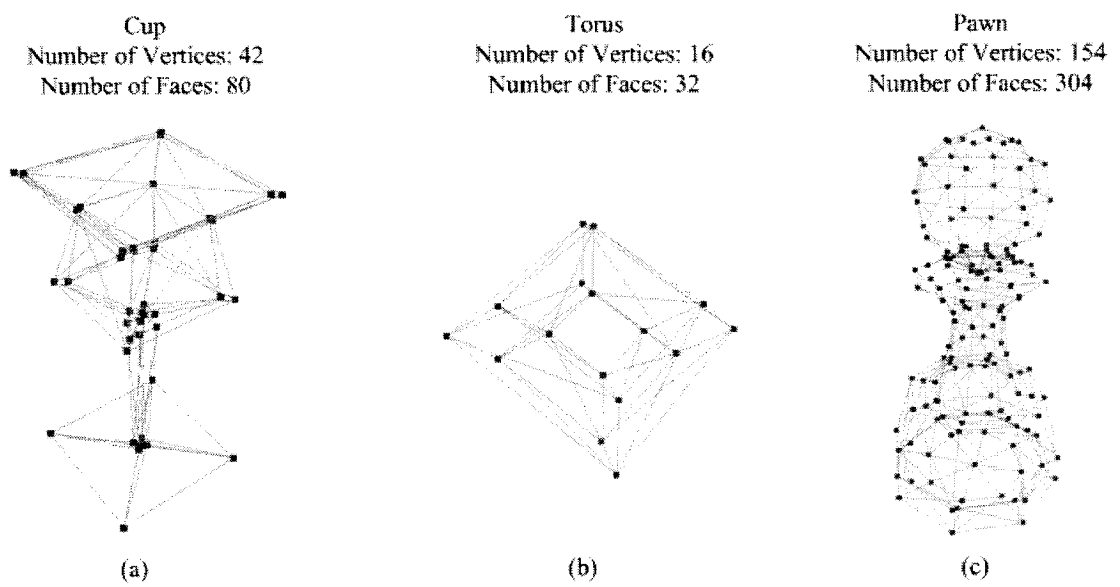


Figure 4-1: The original control meshes of cup, torus and pawn.

Our test begins with inputting the cup control mesh as the initial surface mesh at level 0. Next, we subdivide the initial mesh 6 times by using our editing system and we receive the subdivision surface meshes from level 1 to level 6.

The details about numbers of vertices and faces on the cup control meshes from level 0 to level 6 are listed in Table 4-1:

Table 4-1: The numbers of vertices and faces on the hierarchical subdivision-based cup surface meshes.

Hierarchical cup subdivision control meshes							
Number \ Level	Level 0	Level 1	Level 2	Level 3	Level 4	Level 5	Level 6
Number of Vertices	42	162	642	2562	10242	40962	163842
Number of Faces	80	320	1280	5120	20480	81920	327680

We subdivide the same initial mesh by respectively using Loop subdivision and Modified Butterfly subdivision. For each subdivision, we repeatedly measure n times at each subdivision level, while taking three types of time costs into account: computing mesh connectivity, computing surface normals and performing subdivision. Next we calculate all the average time costs for each subdivision level. Using Equation 4.1, each standard deviation σ of the corresponding average time cost at each subdivision level is also calculated.

$$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2} \tag{4.1}$$

In our project, we set n to 30. For convenience, we list in Table 4-2 all notations used in the following sections.

Table 4-2: The notations of corresponding expressions.

Time cost	The test at i^{th} time ($i \in N$)	Average time cost	Standard deviation	Deviation rate
Compute mesh connectivity	c_i	$\bar{c}$	σ_c	$\triangle c = \sigma_c / \bar{c}$
Compute surface normals	n_i	$\bar{n}$	σ_n	$\triangle n = \sigma_n / \bar{n}$
Perform subdivision	s_i	$\bar{s}$	σ_s	$\triangle s = \sigma_s / \bar{s}$

For performing Loop subdivision on the cup surface meshes, all the average time costs along with their corresponding standard deviations at each level are shown in Table 4-3.

Table 4-3: The corresponding calculation values on the Loop subdivision cup meshes.

	Level 1	Level 2	Level 3	Level 4	Level 5	Level 6
$\bar{c}$	0.005698	0.027268	0.113051	0.427082	1.899569	10.920116
σ_c	0.000231	0.000635	0.001369	0.004884	0.034491	0.114922
$\bar{n}$	0.000164	0.000642	0.002696	0.012062	0.048453	0.195849
σ_n	0.000010	0.000007	0.000033	0.000236	0.000493	0.000876
$\bar{s}$	0.000480	0.002590	0.014398	0.051623	0.208039	0.820351
σ_s	0.000084	0.000113	0.000591	0.000765	0.000574	0.001145

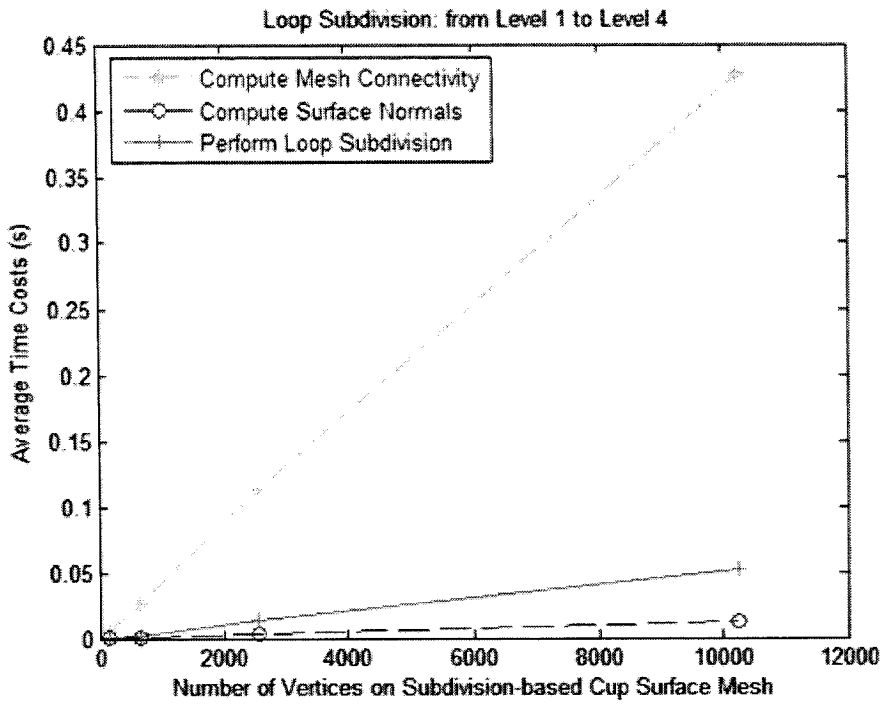
According to Table 4-3, we can obverse that from level 1 to level 6, the threshold of $\triangle c$ is [0.0105, 0.0405], the threshold of $\triangle n$ is [0.0045, 0.0610] and the threshold of $\triangle s$ is [0.0014, 0.1750], which imply that the measured errors are admissible. All corresponding values of $\triangle c$, $\triangle n$ and $\triangle s$ are so small that even using the Matlab's function *errorbar( )*, it is not clear to display their difference. Thus we list the concrete deviation rates of Table 4-3 in Table I-1 (see Appendix I).

We illustrate all the average time costs respectively in two ways: one way is to display the values from level 1 to level 4 (see Figure 4-2 (a)) and another way is to display the values from level 1 to level 6 (see Figure 4-2 (b)).

From Figure 4-2, we can observe that the time cost from level i to level $i+1$ scales almost linearly while the number of vertices on Loop subdivision-based surface meshes at the relevant subdivision level increases. Thus, we conclude that each time when Loop subdivision is performed, the number of vertices on surface meshes is increased, which directly and linearly leads to the increase of time costs for performing Loop subdivision, computing mesh connectivity and surface normals.

We carry out the same evaluation process on the hierarchical Modified Butterfly subdivision cup surfaces to obtain all average time costs with their corresponding standard deviations (see Table 4-4).

(a)



(b)

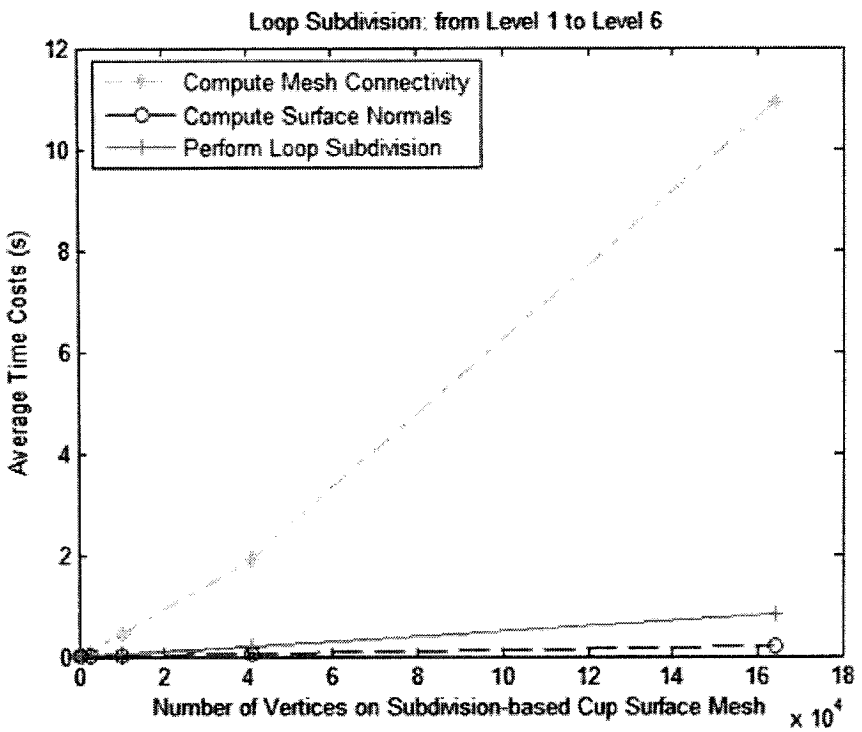


Figure 4-2: Three types of average time costs on the Loop subdivision-based cup surface meshes from level 1 to level 4 (a) or from level 1 to level 6(b).

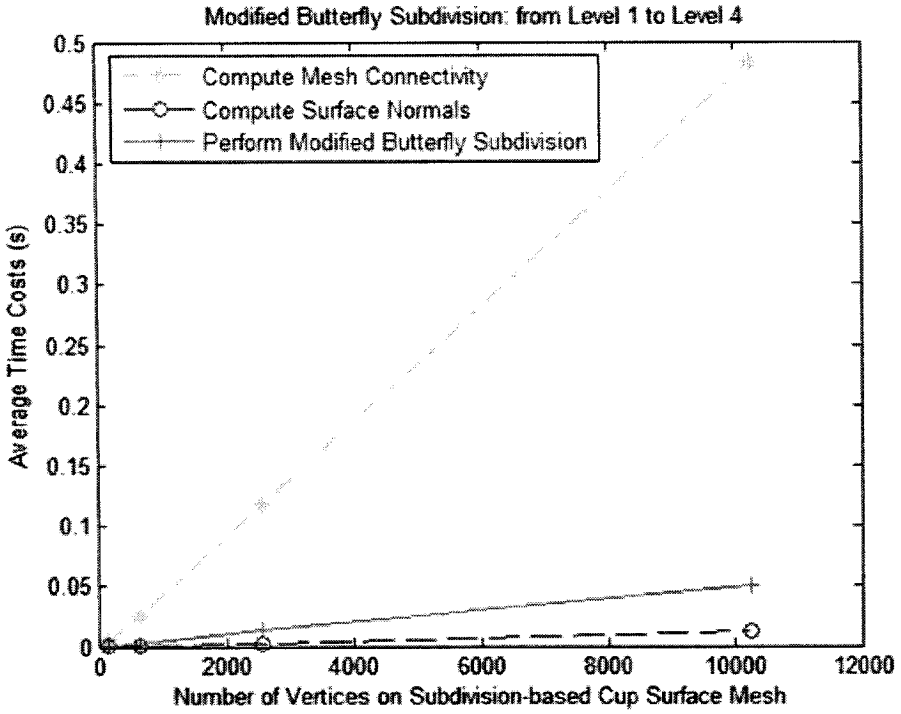
Table 4-4: The corresponding calculation values on the Modified Butterfly subdivision-based cup meshes.

	Level 1	Level 2	Level 3	Level 4	Level 5	Level 6
$\bar{c}$	0.005574	0.025952	0.118392	0.482986	1.998162	11.443304
σ_c	0.000215	0.000520	0.001607	0.003194	0.020734	0.143320
$\bar{n}$	0.000165	0.000651	0.002694	0.012011	0.048314	0.194825
σ_n	0.000007	0.000014	0.000018	0.000152	0.000450	0.001034
$\bar{s}$	0.000568	0.002714	0.014097	0.049838	0.212584	0.831132
σ_s	0.000035	0.000068	0.000414	0.000720	0.002595	0.040283

The concrete deviation rates of Table 4-3 are listed in Table I-1 (see Appendix I), which prove that the measured errors are also admissible. All the relevant average time costs in Table 4-4 are illustrated in Figure 4-3.

From Table 4-3 and Table 4-4, we can observe that for Loop subdivision and Modified Butterfly subdivision, time costs of performing subdivision are nearly the same, and time costs of computing mesh connectivity and surface normals are also very close. Besides, Figure 4-3 illustrates that time costs of performing Modified Butterfly subdivision increase linearly as the number of vertices increases on subdivision surfaces from lower level to higher level. Thus we can conclude that choosing Loop subdivision or Modified Butterfly subdivision is not the factor which influences global time costs of subdivision.

(a)



(b)

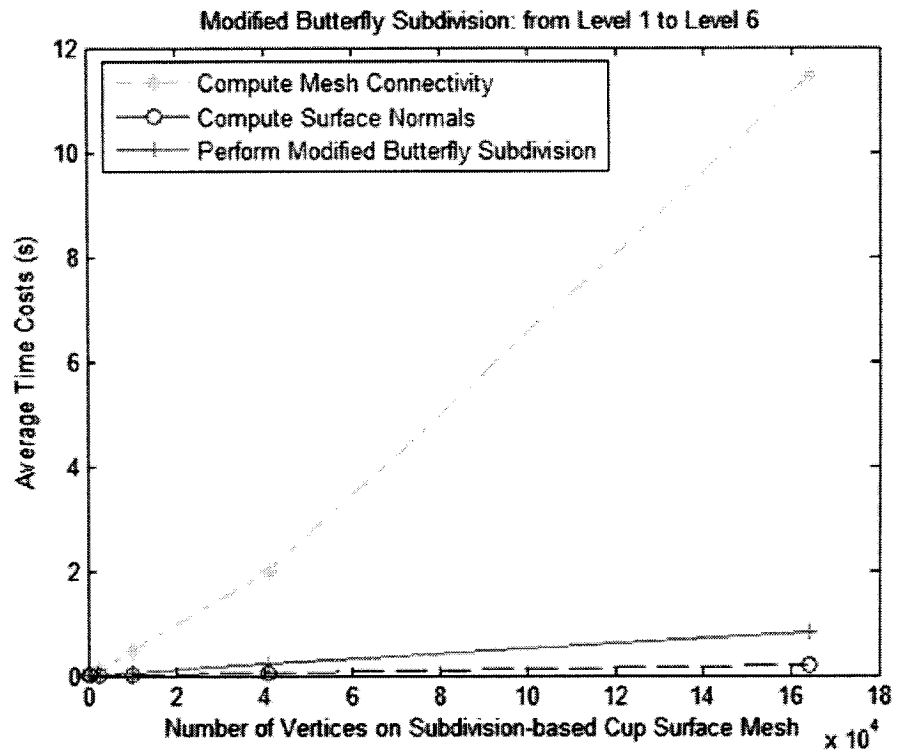


Figure 4-3: Three types of average time costs on the Modified Butterfly subdivision cup meshes from level 1 to level 4 (a) or from level 1 to level 6(b).

In order to verify whether other examples also satisfy the conclusion which we discussed above, we take another two examples into consideration. One example is a torus control mesh (see Figure 4-1(b)) and another one is a pawn control mesh (see Figure 4-1(c)). Comparing with the original cup control mesh, there are less vertices and faces on the original torus control mesh, while there are much more vertices and faces on the original pawn control mesh. Besides, the torus and the pawn surface meshes are closed.

Accordingly, we list the numbers of vertices on torus and pawn surface meshes from level 1 to level 3 in Table 4-5. The concrete average time costs with their corresponding standard deviations on torus or pawn surface are respectively listed in Table 4-6 (torus surface) and Table 4-7 (pawn surface).

Table 4-5: The Number of vertices and the number of faces on the torus or pawn subdivision-based surface meshes from level 1 to level 3.

		Level 1	Level 2	Level 3
Torus	Number of Vertex	64	256	1024
	Number of Faces	128	512	2048
Pawn	Number of Vertex	610	2434	9730
	Number of Faces	1216	4864	19456

Table 4-6: The corresponding calculation values of performing Loop or Modified Butterfly subdivision on the torus meshes.

	Loop Subdivision			Modified Butterfly Subdivision		
	Level 1	Level 2	Level 3	Level 1	Level 2	Level 3
$\bar{c}$	0.002054	0.010802	0.044989	0.002038	0.010301	0.047898
σ_c	0.000065	0.000431	0.000709	0.000235	0.000269	0.001109
$\bar{n}$	0.000322	0.001286	0.005235	0.000326	0.001294	0.005267
σ_n	0.000000	0.000006	0.000018	0.000013	0.000049	0.000170
$\bar{s}$	0.000183	0.001087	0.005619	0.000178	0.001023	0.005998
σ_s	0.000008	0.000028	0.000223	0.000013	0.000038	0.000704

Table 4-7: The corresponding calculation values of performing Loop or Modified Butterfly subdivision on the pawn meshes.

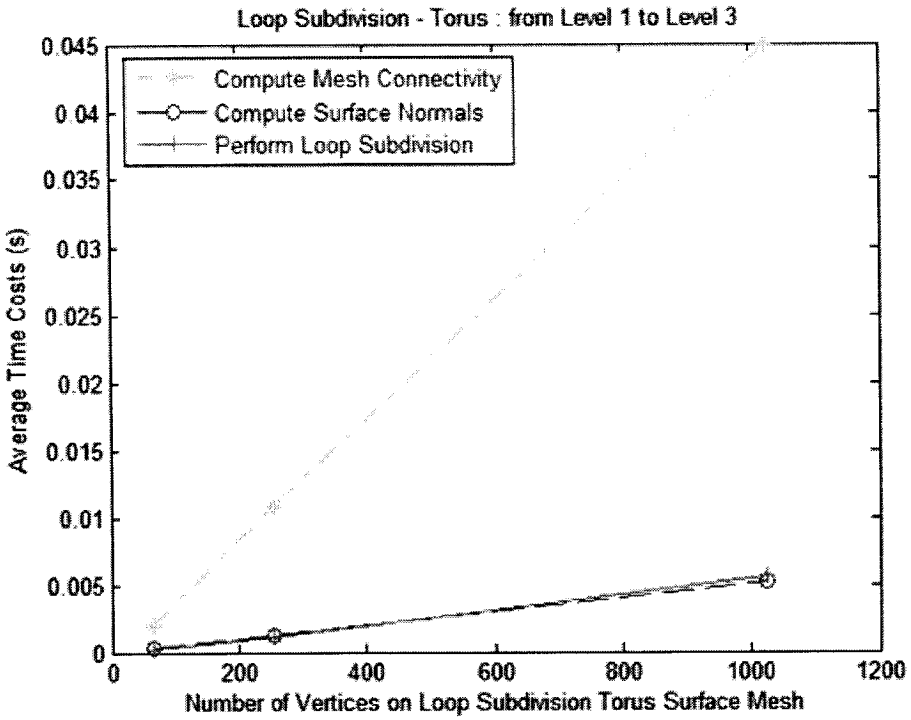
	Loop Subdivision			Modified Butterfly Subdivision		
	Level 1	Level 2	Level 3	Level 1	Level 2	Level 3
$\bar{c}$	0.025869	0.101874	0.441695	0.026891	0.106453	0.456082
σ_c	0.001311	0.001127	0.004045	0.001219	0.001506	0.003409
$\bar{n}$	0.000626	0.002530	0.011618	0.000617	0.002531	0.011573
σ_n	0.000032	0.000093	0.000357	0.000005	0.000082	0.000247
$\bar{s}$	0.002371	0.012760	0.048895	0.002665	0.012791	0.049259
σ_s	0.000063	0.000670	0.000565	0.000276	0.000164	0.000573

We illustrate the corresponding evaluation from level 1 to level 3 on the torus surface mesh based on three factors in Figure 4-4. Similarly, the corresponding evaluation on the pawn surface meshes is visualized in Figure 4-5. Figure 4-4 comes from Table 4-6 and Figure 4-4 comes from Table 4-7. The corresponding deviation rates of Table 4-6 and Table 4-7 are separately listed in Table I-3 and Table I-4.

From Figure 4-4 and Figure 4-5, we can observe that even though there are some differences, time costs still increase nearly linearly as the number of vertices on surface mesh increases. Whereas, the results from Table I-3 and Table I-4 guarantee these differences can be viewed as admissible errors.

According to all above results, we can observe that computing mesh connectivity consumes most of system time. Meanwhile, we can observe that performing Loop subdivision costs nearly the same time as performing Modified Butterfly subdivision.

(a)



(b)

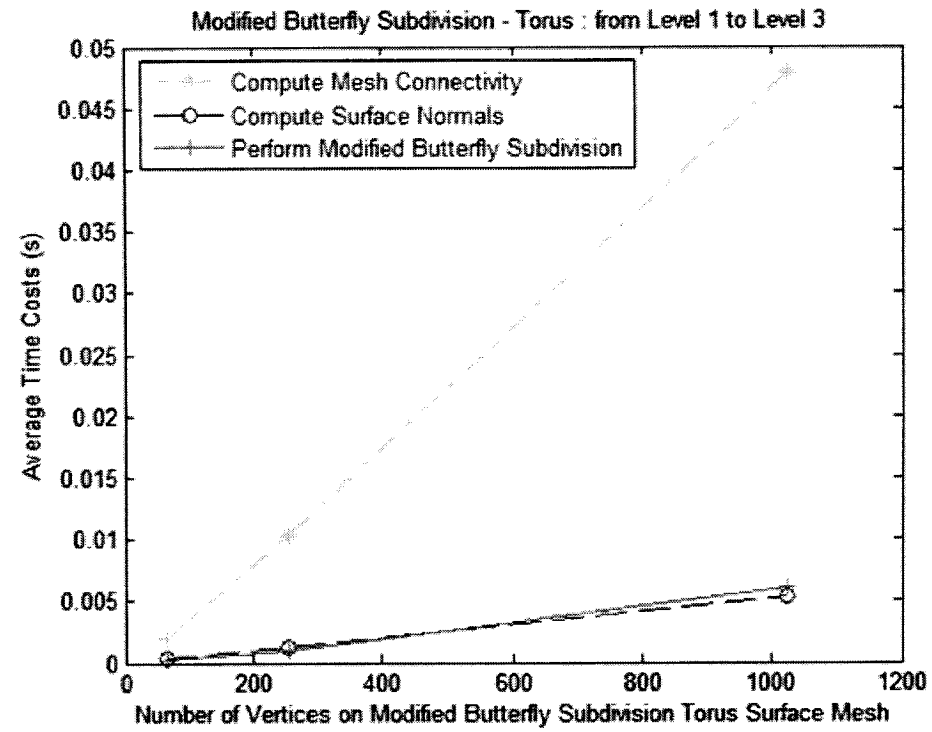
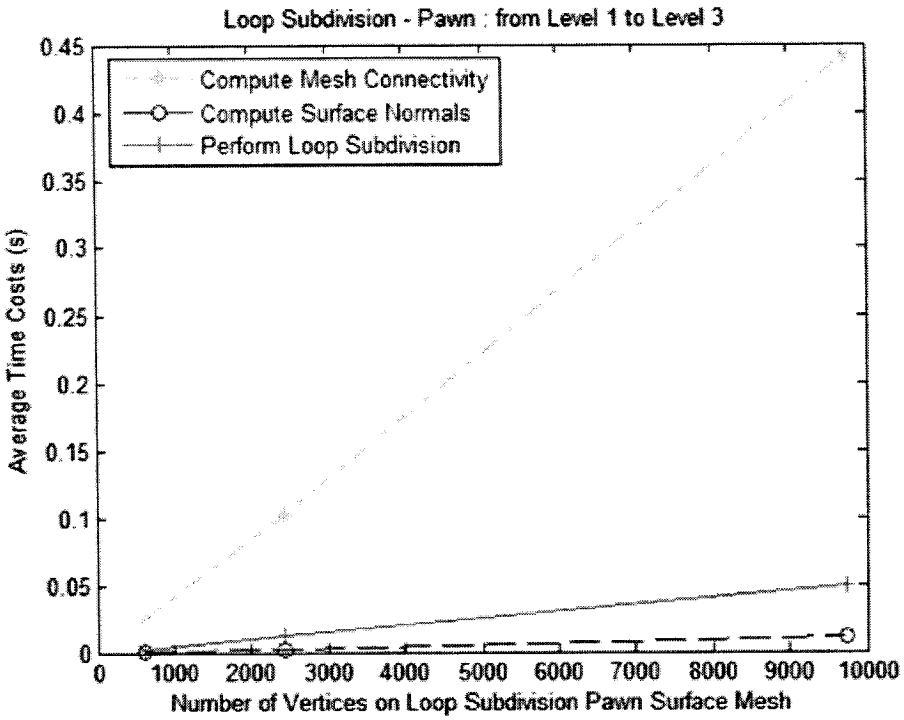


Figure 4-4: Evaluation of three factors on the torus control meshes from level 1 to level 3: (a) Loop subdivision and (b) Modified Butterfly subdivision.

(a)



(b)

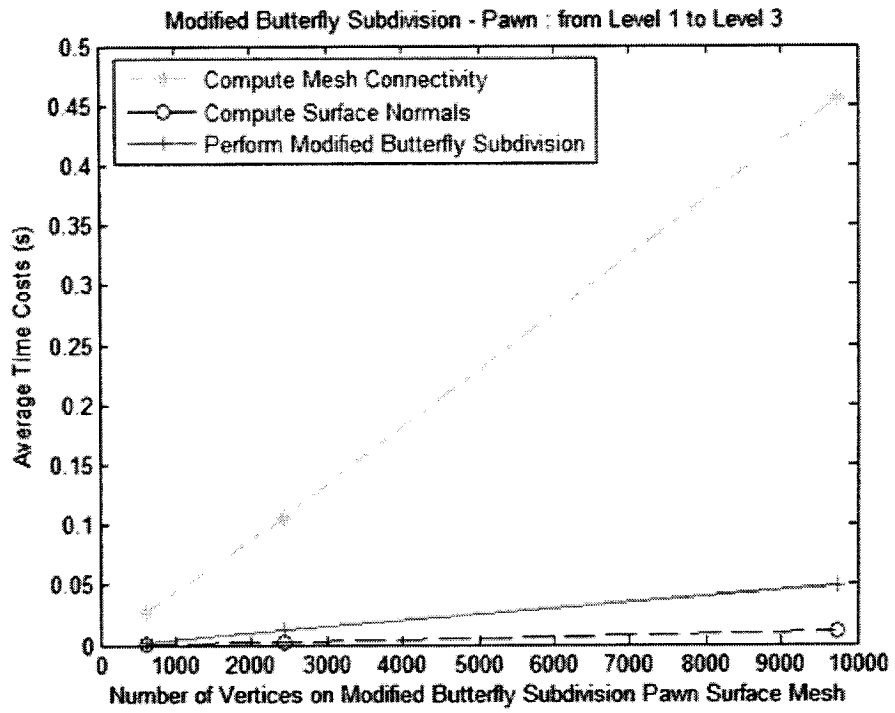


Figure 4-5: Evaluation of three factors on the pawn control meshes from level 1 to level 3: (a) Loop subdivision and (b) Modified Butterfly subdivision.

4.2 Validity of Implementing Local Support

As we have discussed in section 3.2, when one selected vertex is modified interactively on Loop surface meshes, local support is implemented in our editing system to efficiently recalculate new positions of its neighbouring vertices.

In our program, we have written two Loop subdivision functions, one with local support and the other one without local support. Our tests analyze the differences between performing the same test examples by using these two functions. The tests aim to record and compare the time costs of editing vertices on subdivision surface meshes, while calling the two different subdivision functions.

Our tests will only consider moving one vertex to an arbitrary location without accurately setting any limitation of its position. Because when one vertex on a surface is being modified, the distance between its new position and its former position causes nearly no difference on the time needed to perform subdivision.

We still use the cup (see Figure 4-6 (a)), torus (see Figure 4-6 (b)) and pawn (see Figure 4-6 (c)) surfaces as our test examples. Their corresponding original control meshes and the Loop subdivision surfaces at level 3 with the selected vertex for editing are shown in Figure 4-6. The ones on the left are

without editing and the ones on the right are with editing.

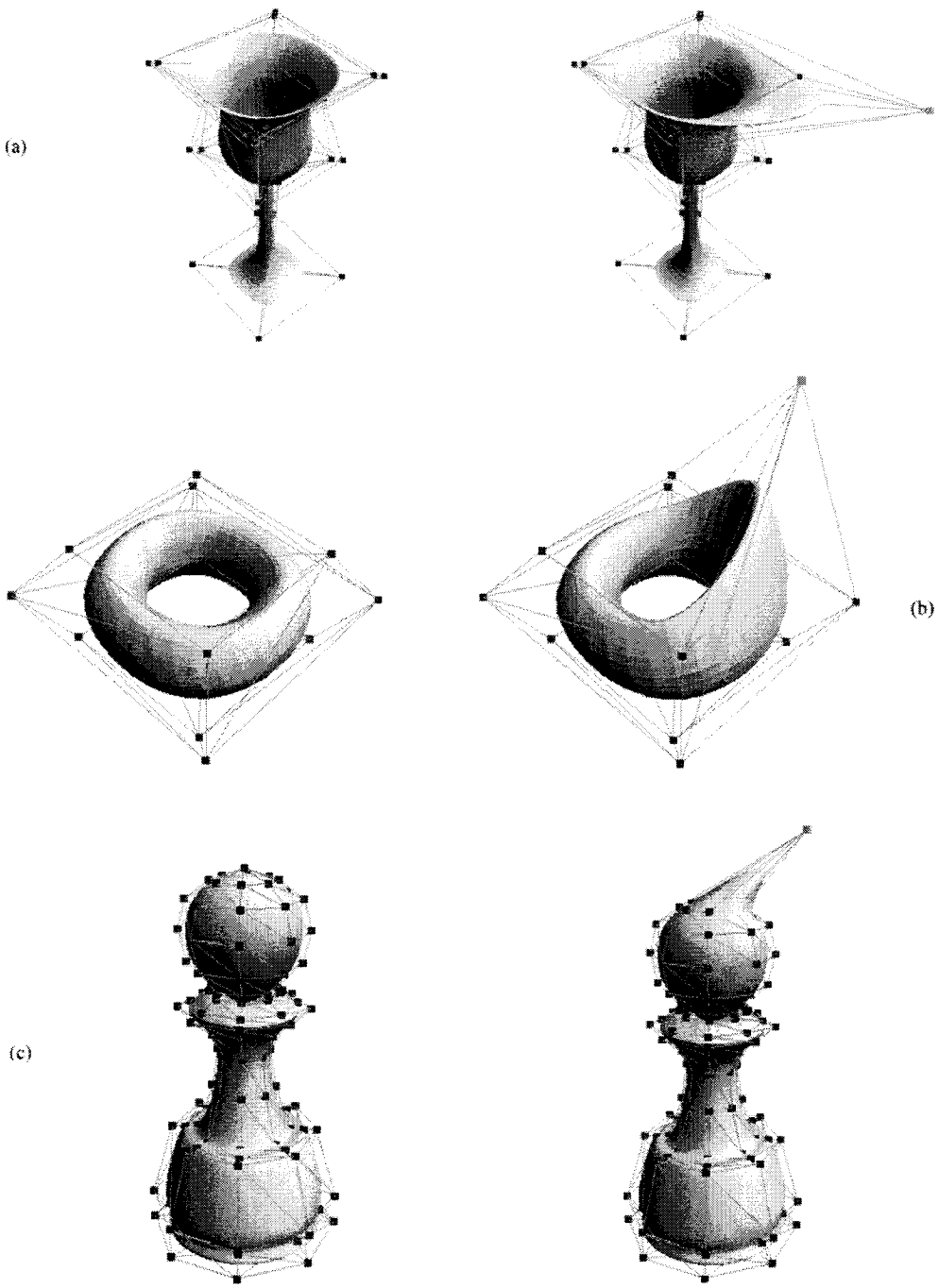


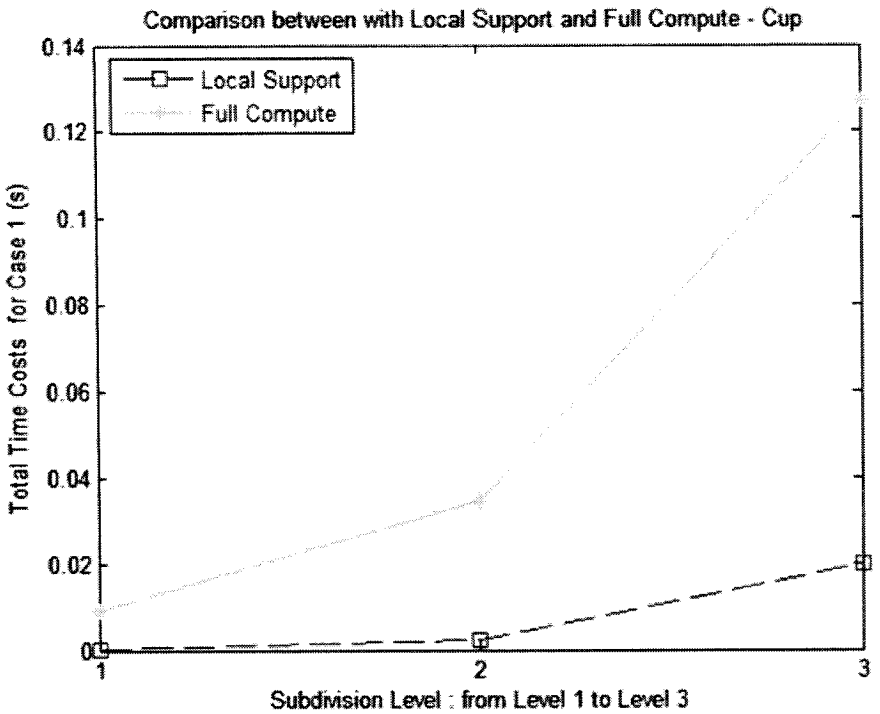
Figure 4-6: Illustration of modifying one selected vertex on Loop subdivision-based (a) cup, (b) torus and (c) pawn surface meshes.

Because even though the difference caused by moving vertex can be ignored, it is difficult to guarantee each time when the same vertex on the same initial surface is being moved, there is absolutely no difference. Thus, for each test example, we do a random sample test twice and we present them in two cases. For each case, we calculate total time costs of performing Loop subdivision, computing mesh connectivity and surface normals. We illustrate the total time cost differences between using local support and without local support (full compute) by performing Loop subdivision from level 1 to level 3 on the cup (see Figure 4-7), torus (see Figure 4-8), and pawn (see Figure 4-9) surfaces.

With local support, mesh connectivity is not recalculated each time when there are some changes on surface meshes. According to Figure 4-7, Figure 4-8 and Figure 4-9, we can observe that the time costs using local support are much less than the time costs using full compute.

In Figure 4-10, we illustrate the economized time costs using local support on subdivision-based cup, torus and pawn surface from level 1 to level 3. From this illustration, we can observe that from lower subdivision level to higher subdivision level, the economized time costs on each surface mesh increase. Besides, we can observe that pawn subdivision surface with the densest mesh economizes most time by using local support, whereas torus surface with the sparsest mesh economizes least time by using local support. Thus we can conclude that as subdivision level goes higher and as the complexity or density surface mesh increases, the economized time cost using local support also increases.

Case 1



Case 2

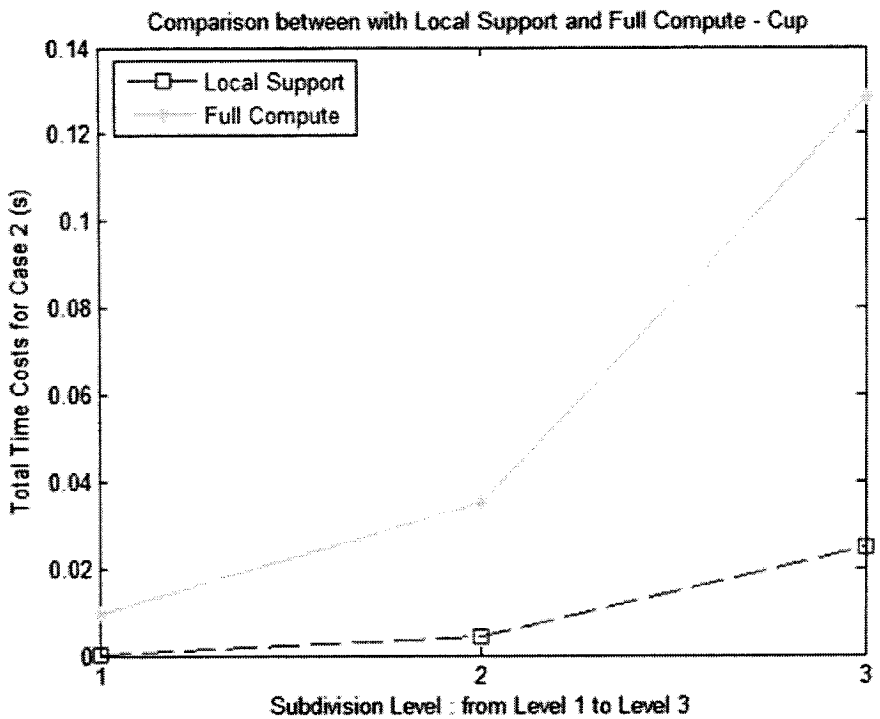
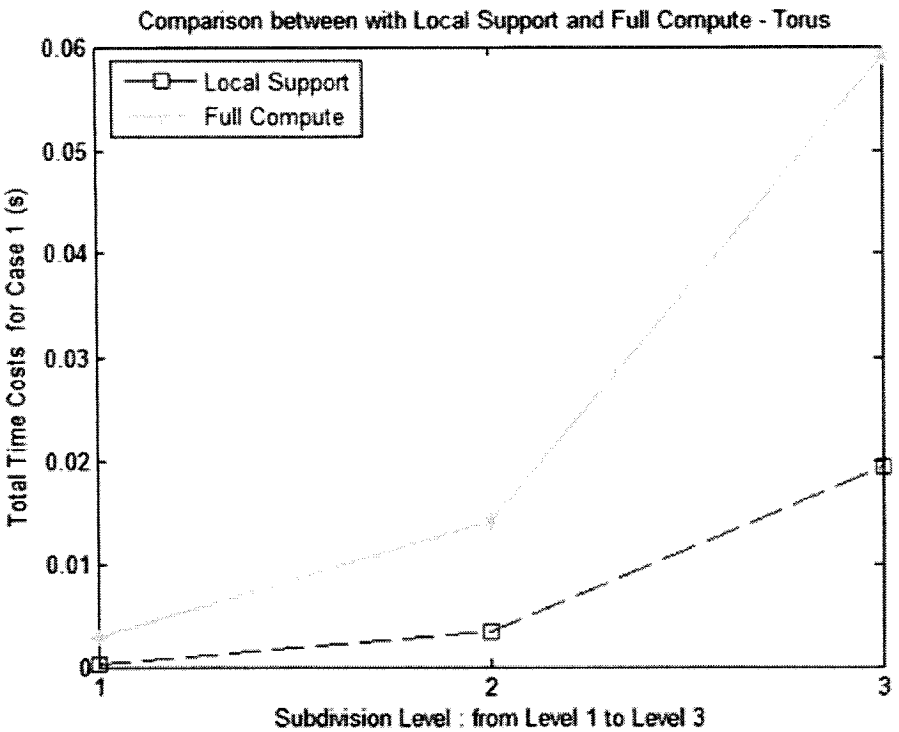


Figure 4-7: The comparison of editing one selected vertex on the Loop subdivision cup surfaces between with local support and with full compute.

Case 1



Case 2

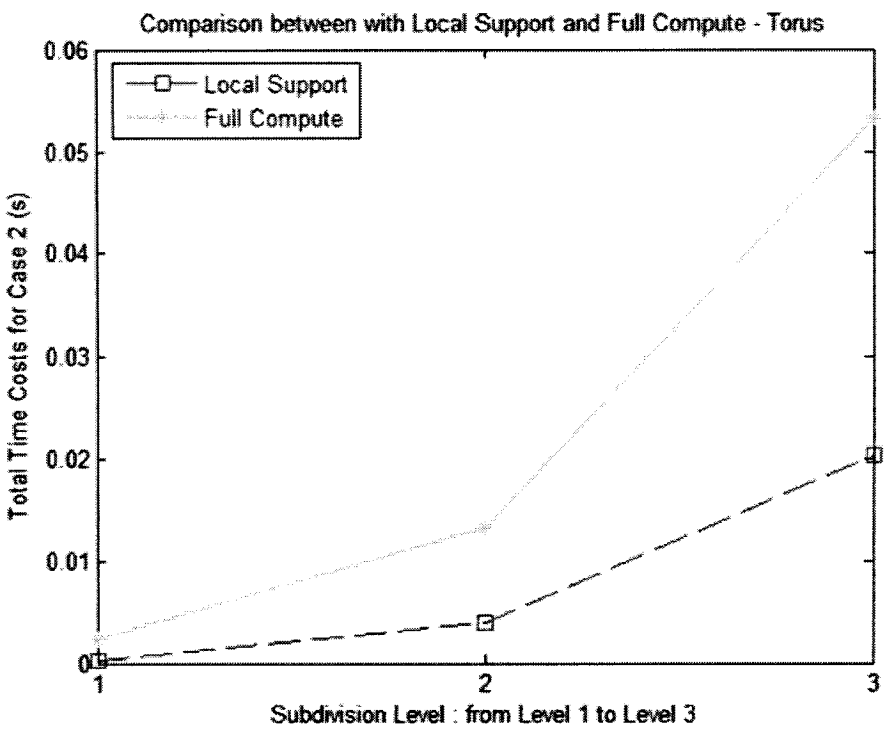
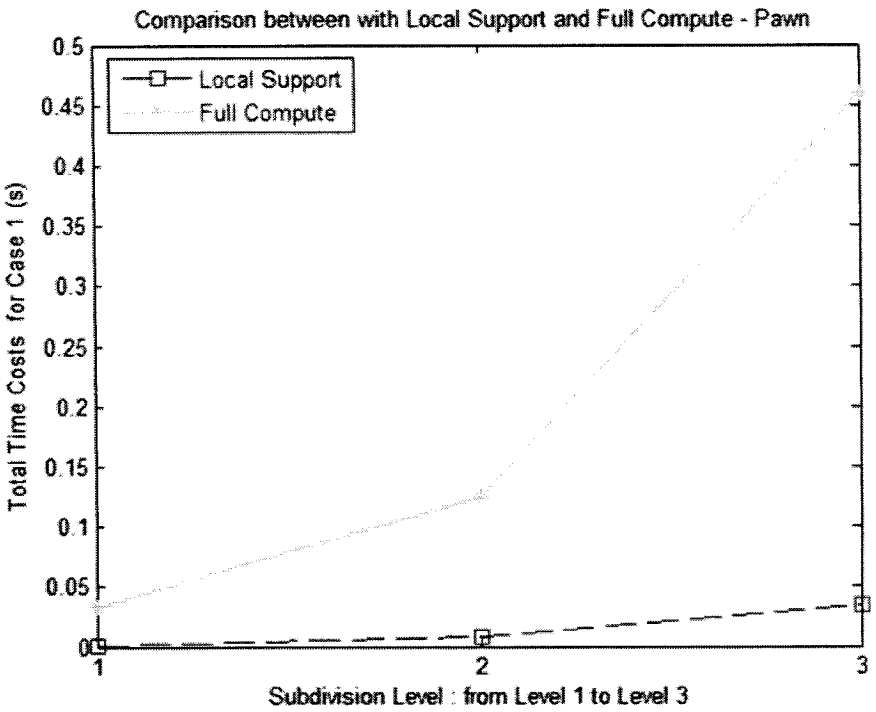


Figure 4-8: The comparison of editing one selected vertex on the Loop subdivision torus surfaces between with local support and with full compute.

Case 1



Case 2

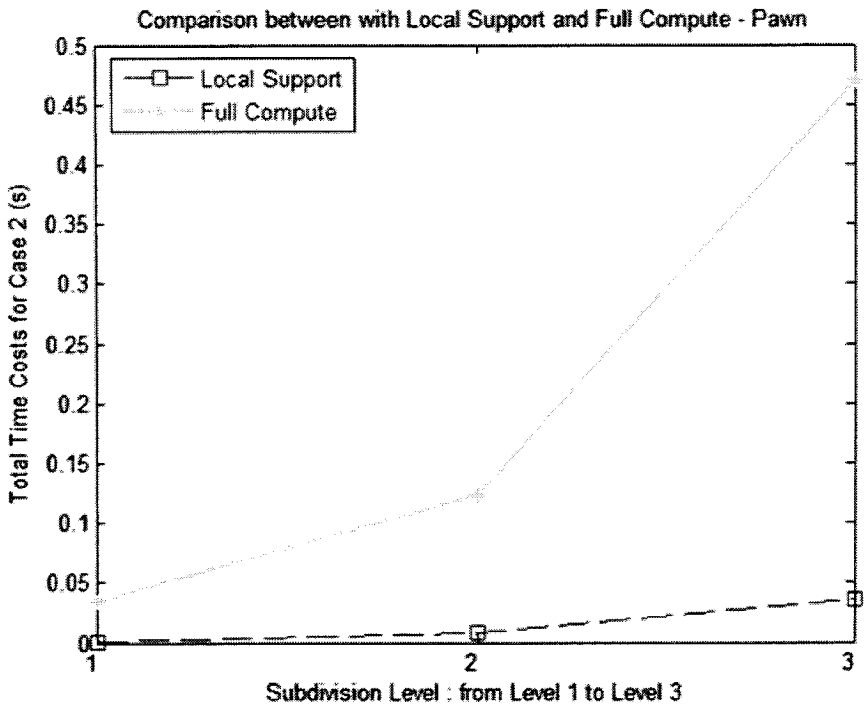
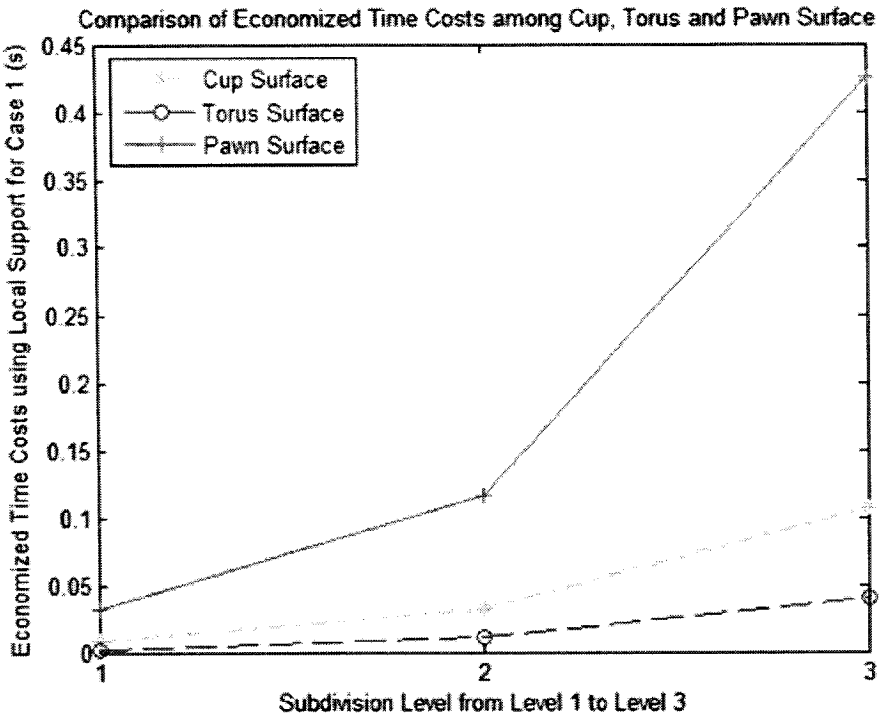


Figure 4-9: The comparison of editing one selected vertex on the Loop subdivision pawn surfaces between with local support and with full compute.

Case 1



Case 2

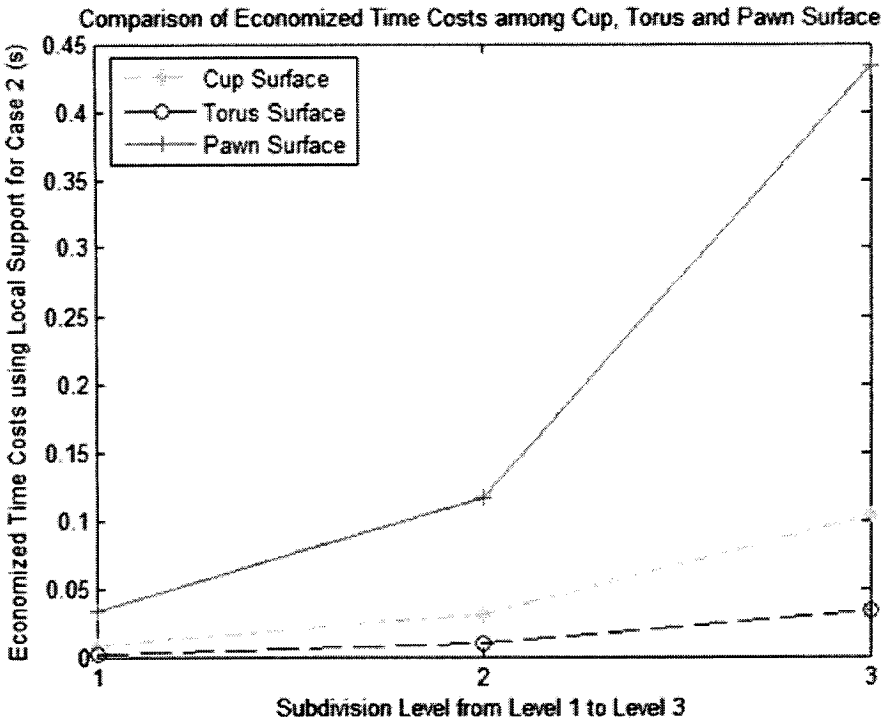


Figure 4-10: The comparison of economized time costs using local support on the cup, torus and pawn surfaces from level 1 to level 3.

4.3 Validity of Preserving Changes during Subdivision

In section 3.2.3, we mentioned the problem of keeping changes from higher subdivision levels while performing subdivision. In section 3.3.2, we proposed an algorithm that is implemented in our editing system to solve this problem. Because the interactive editing on surface meshes is a real-time process, we show how changes may be preserved during subdivision by illustrating the concrete editing procedure of one example. We choose one 3D plane (see Figure 4-11) as input, and then we edit their subdivision surfaces at different levels to demonstrate how changes from higher level are kept.

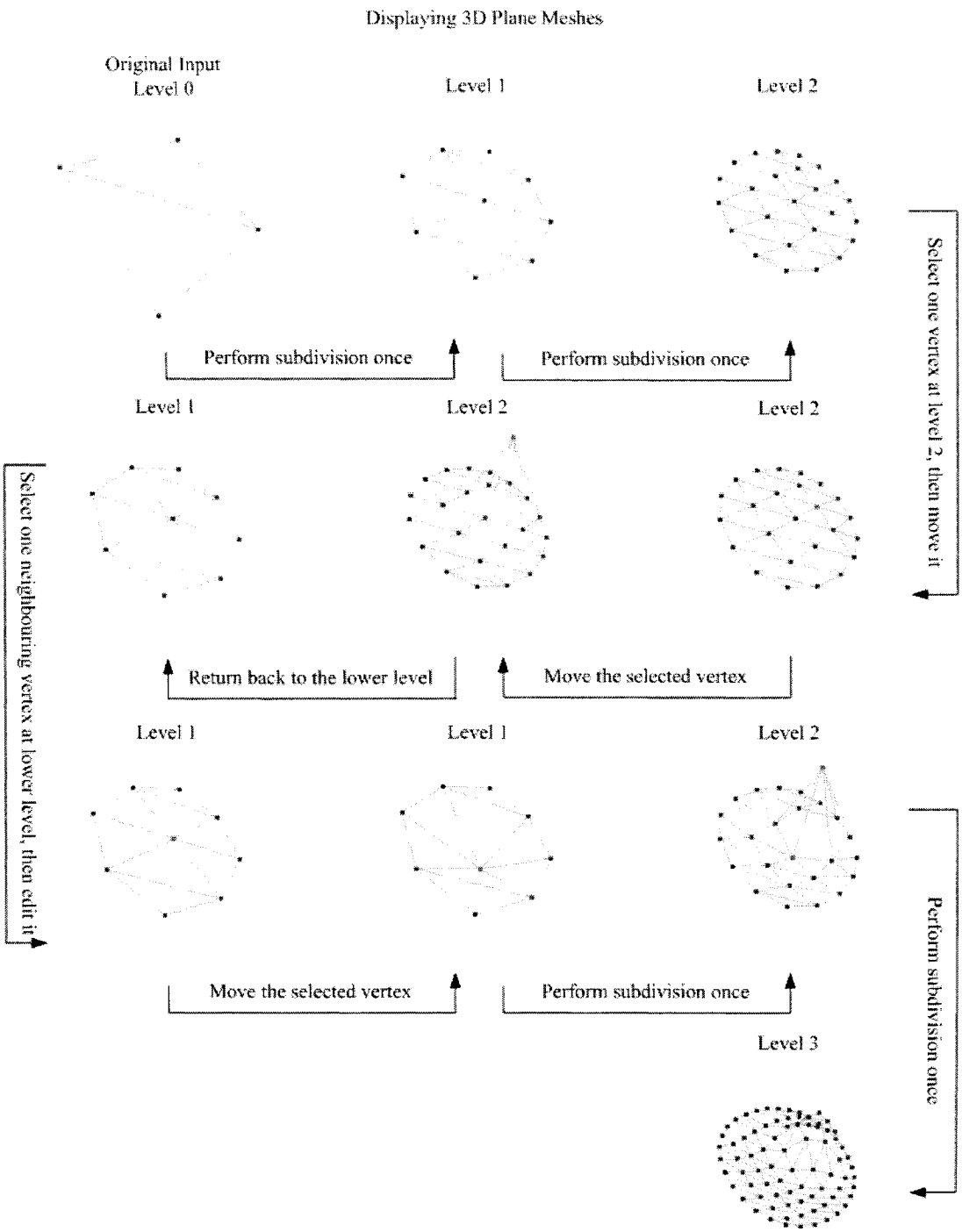


Figure 4-11: The illustration of editing 3D plane surface meshes at multi-levels.

4.4 Interactive Editing on Subdivision-based Surfaces

Keeping sharp features on subdivision-based surface meshes is another important objective for our project. We implement vertex crease and edge crease for Loop subdivision and edge crease for Modified Butterfly subdivision. We demonstrate their results in the following figures.

In Figure 4-12, the ones on the second row are defined with vertex creases, where at the top, the four vertices are selected. The ones on the third row are defined with edge creases, where four edges on top are selected.

In Figure 4-13, Modified Butterfly subdivision is used to generate subdivision-based surface meshes. On Modified Butterfly subdivision-based surface meshes, the four edges are defined as edge crease.

In Figure 4-14, we subdivide the cup control mesh (see Figure 4-1 (a)) once by performing Loop subdivision. Then we use loop subdivision-based cup surface as the original input mesh for editing. And then we define edge creases around the bottom line on the cup surface, while performing Modified Butterfly subdivision on its input mesh. The last result is illustrated in Figure 4-14.

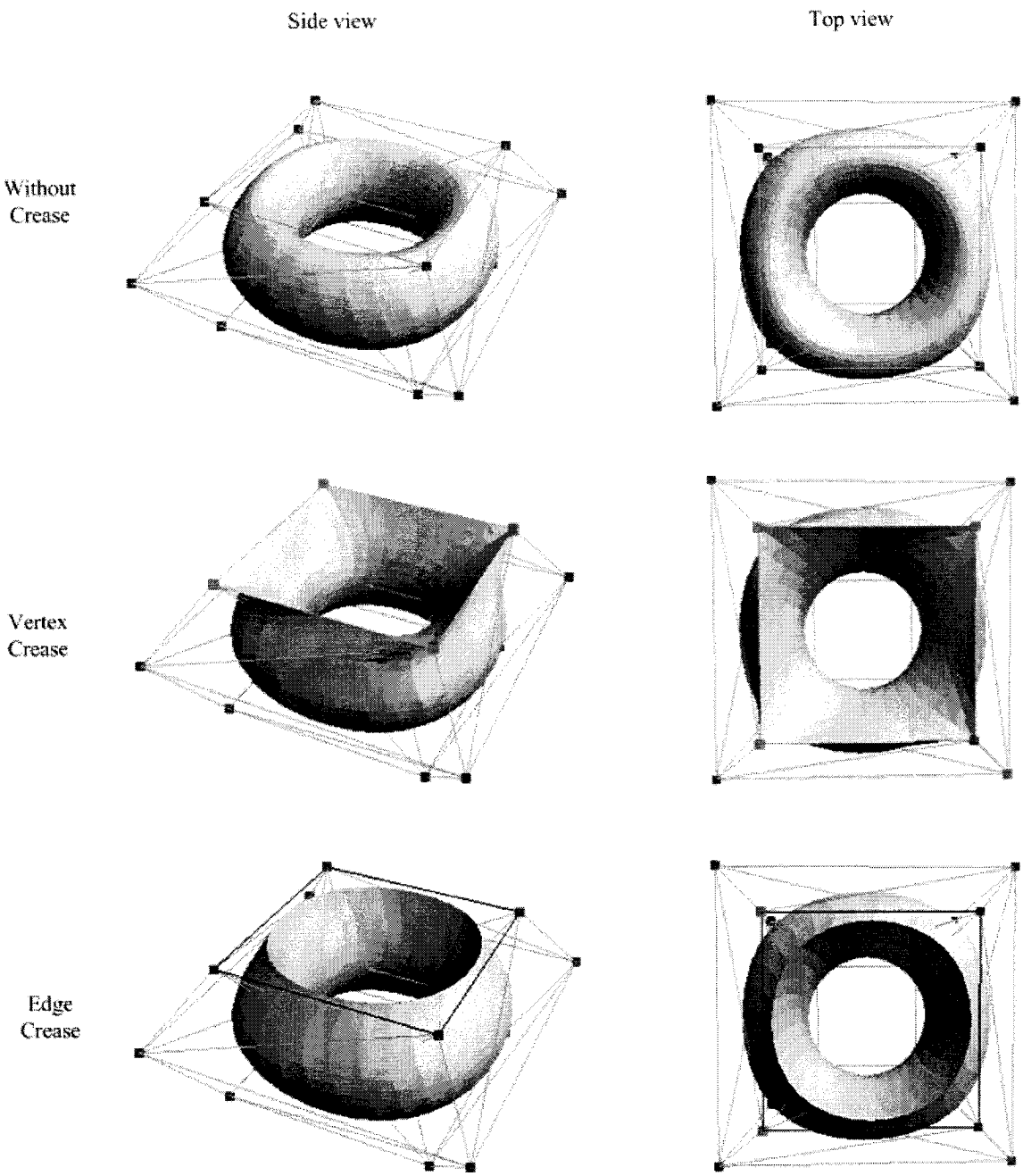


Figure 4-12: The illustration of defining vertex creases and edge creases on the top of Loop subdivision torus surface meshes, where the surfaces inside the control mesh are subdivided 3 times from two different points of views.

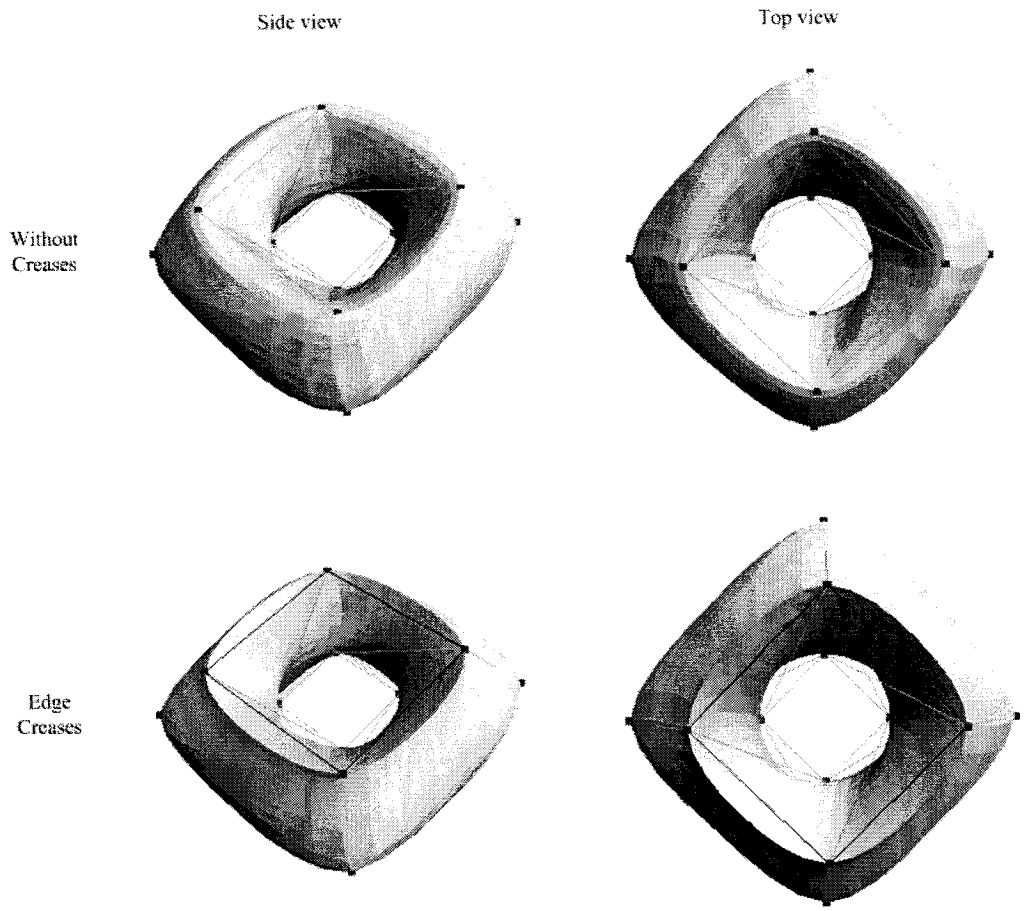


Figure 4-13: Define edge crease on the top of Modified Butterfly subdivision torus surface meshes, where the surfaces inside the control mesh are subdivided 3 times with two different points of view.

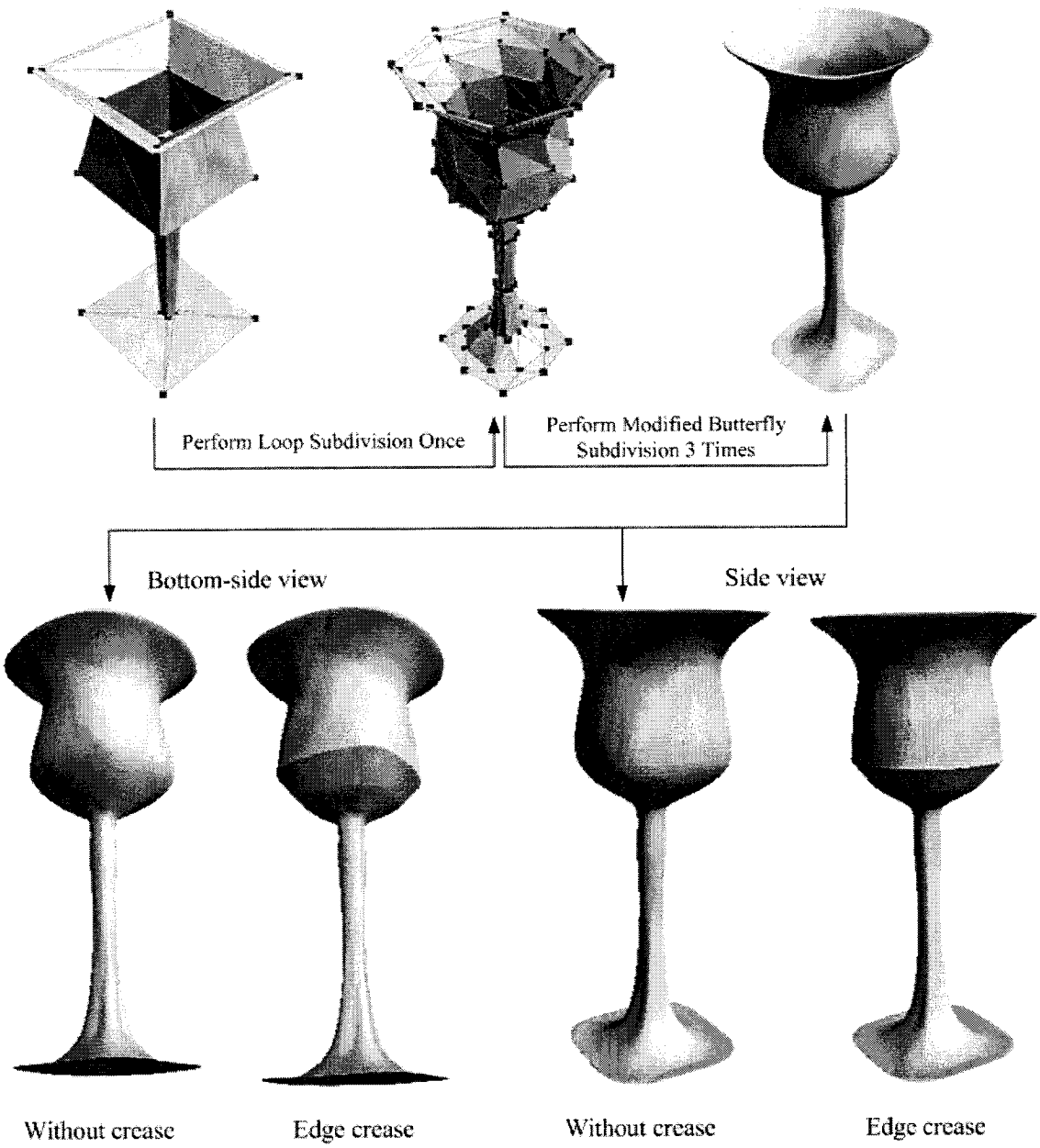


Figure 4-14: Define edge crease on the bottom line of the subdivision-based cup surface.

CHAPTER 5

CONCLUSION AND FUTURE WORK

This thesis addresses the issues of interactive editing on multi-resolution subdivision-based surface meshes. Our project presents an editing system which can interactively edit surface meshes. After the relevant implementation, we evaluate system performance of the editing system, which is proved to achieve the objective of our work. In this chapter, we summarize our work and explore future work.

5.1 Summary

At the beginning of our thesis, we provided an introduction of related works at length and categorize them into four topics: surface representation, subdivision surfaces, multi-resolution technique with LOD queries and interactive mesh editing. We described their fundamental theories, presented some useful applications in certain domains, discussed their challenges and addressed their relevant innovations and approaches. We presented an extensive overview on valuable references, which particularly inspired our work.

Then we exploited the methods of implementing one approximating subdivision scheme: Loop subdivision and one interpolating subdivision scheme: Modified Butterfly subdivision as the two basic subdivision schemes to generate subdivision surface meshes. Our work distinguished from the other related works by providing a comprehensive comparison between these two subdivision schemes and addressing the effective incorporation between two different subdivisions.

What does bring our work forward is to create creases on two kinds of subdivision-based surfaces. On the one hand, we set up two types of creases creation, vertex crease and edge crease on Loop subdivision surfaces. In the case of edge creases, we creatively appended some specific rules of calculating two or more than two creases jointing together. On the other hand, we particularly updated the masks for boundaries or creases for Modified Butterfly subdivision surfaces.

In order to enhance the intuitive specification of subdivision surface, we improved the algorithm in our interactive editing system. Thus, another innovation of our work lies in building the algorithm solving the problem of keeping all changes on subdivision-generated control meshes even from higher subdivision levels, while still guarantying finite support.

Finally, we added new practical functions to evaluate system performance. Through these functions, we obtained concrete statistics that we used to verify our work.

5.2 Future Work

Without doubt, interactive editing of multi-resolution subdivision-based surface meshes is an active cross-discipline research area, where enlightening issues and innovative ideas always take place. As our research is developing, we also realize that related works cover pretty extensive opportunities. Therefore, in this section we try to outlook the possible opportunities of future research.

1. One Possible Opportunity in Subdivision

When massive original control meshes are input or rendered, certain features or a few local parts on control meshes are not necessary to get finer or smoother. Adaptive subdivision only calculates and subdivides the parts requiring more details, instead of costing more time and more memory to refine all surfaces. Thus, the research on adaptive subdivision holds some favourable prospects.

2. Possible Opportunities in Multi-resolution

Mesh simplification can be introduced more widely in the multi-resolution framework. To evaluate surface quality, mesh simplification potentially requires establishing more measure metrics.

Moreover, due to the results from Chapter 4, we realize that we can ameliorate the algorithm for calculating mesh connectivity within the multi-resolution framework. Comparing with time used to perform subdivision and calculate surface normals, much more time is spent in computing mesh connectivity. If each time when control mesh is initialized, we only update the changes from last time instead of recalculating all the information on the surface, the system will use much less time as a whole for the subdivision procedure.

3. Possible Opportunities in Mesh Editing

The manipulation methods on multi-resolution surface mesh can be improved to be more intuitive. The editing operations can be extended in the future. The sharp features of subdivision surfaces also hold their potentials to be better exhibited.

4. One Possible Opportunity in Evaluation

In recent years, the area of evaluating subdivision surfaces has become very active. Novel algorithms requiring less memory are constantly developed [Bolz and Schroder 2002, Bischoff et al. 2000]. The knowledge about analytic evaluation is also continuously systematized [Hall 2001]. Thus, the evaluation methods can be integrated in our work as one future possibility.

REFERENCES

ALLIEZ, P., GOTSMAN, C. 2004. *Recent Advances in Compression of 3D Meshes*. Published in the book "Advances in Multiresolution for Geometric Modelling". Springer. 2004.
http://www.cs.technion.ac.il/~gotsman/AmendedPubl/Pierre/compression_survey.pdf

ALLIEZ, P., UCELLI, G., GOTSMAN, C., ATTENE, M. 2005. *Recent Advances in Remeshing of Surfaces*. Springer. 2005.
http://www.cs.technion.ac.il/~gotsman/AmendedPubl/Pierre/remeshing_survey.pdf

ASPERT, N., SANTA-CRUZ, D., EBRAHIMI, T. 2002. *MESH: Measuring Error between Surfaces using the Hausdorff Distance*. Proceedings of the IEEE International Conference on Multimedia and Expo 2002 (ICME), Volume. I. Pages: 705-708. 2002.
http://mesh.berlios.de/mesh.pdf
http://mesh.berlios.de/

ATTENE, M., FALCIDIENO, B., ROSSIGNAC, J., SPAGNUOLO, M. 2005. *Sharpen&Bend: Recovering Curved Sharp Edges in Triangle Meshes Produced by Feature-Insensitive Sampling*. Visualization and Computer Graphics, IEEE Transactions on Volume 11, Issue 2. Pages: 181-192. March-April 2005.

http://ieeexplore.ieee.org/search/wrapper.jsp?arnumber=1388229

BAJAJ, C.L., PASCUCCI, V., ZHUANG, G.Z. 1999. *Progressive Compression and Transmission of Arbitrary Triangular Meshes.* 10th Proceedings of IEEE Visualization 1999 (Vis'99). Pages: 307-316. 1999.
http://doi.ieeecomputersociety.org/10.1109/VISUAL.1999.809902
http://www.pascucci.org/pdf-presentations/vis99.pdf

BIERMANN, H., MARTIN, I., BERNARDINI, F., ZORIN, D. 2002. *Cut-and-Paste Editing of Multiresolution Subdivision Surfaces.* ACM Transactions on Graphics, Volume 21, No. 3, Proceedings of ACM SIGGRAPH. Pages: 312-321. July, 2002.
http://www.research.ibm.com/people/i/imartin/papers/surfpaste-sig02.pdf

BIERMANN, H., MARTIN, I., ZORIN, D., BERNARDINI, F. 2001. *Sharp Features on Multiresolution Subdivision Surfaces.* Proceedings of the 9th Pacific Conference on Computer Graphics and Applications. Pages: 140-149. 2001.
http://mrl.nyu.edu/publications/sharpfeatures/sharpfeatures.pdf

BISCHOFF, S., KOBBELT, L.P., SIEDEL, H.P. 2000. *Towards Hardware Implementation of Loop Subdivision.* SIGGRAPH/Eurographics Workshop on Graphics Hardware. Pages: 41-50. 2000.
http://www-i8.informatik.rwth-aachen.de/publications/downloads/loop_render.pdf

BOLZ, J., SCHRODER, P. 2002. *Evaluation of Subdivision Surfaces on Programmable Graphics Hardware*. Symposium, Web3D 2002. 2002. http://multires.caltech.edu/pubs/GPUSubD.pdf

BOTSCH, M. KOBBELT, L.P. 2003. *Multiresolution Surface Representation Based on Displacement Volumes*. Computer Graphics Forum 22(3) (Eurographics 2003 Proceedings). Pages: 483-491. 2003. http://www-i8.informatik.rwth-aachen.de/publications/downloads/voldetail.pdf

BRICKHILL, D. 2001. *Practical Implementation Techniques for Multi-Resolution Subdivision Surfaces*. The Golden Gate Game Company. 2001. http://www.gdconf.com/archives/2001/brickhilld.doc

BRUNNETT, G., HAMANN, B., MUELLER, H., LINSEN, L. (Eds.) 2004. *Geometric Modeling for Scientific Visualization*. Springer-Verlag, Heidelberg, Germany. Series: Mathematics and Visualization. 488 pages. 2004.

CATMULL, E., CLARK, J. 1978. *Recursively Generated B-Spline Surfaces on Arbitrary Topological Meshes*. Computer Aided Design 10, Volume 6. Pages: 350-355. 1978.

CHAIKIN, G. 1974. *An Algorithm for High Speed Curve Generation*. Computer Graphics and Image Processing 3. Pages: 346-349, 1974.

CIAMPALINI, A., CIGNONI, P., MONTANI, C., SCOPIGNO, R. 1997. *Multiresolution Decimation based on Global Error*. The Visual Computer, Springer International, 13(5). Pages: 228-246. 1997.
http://portal.acm.org/citation.cfm?id=868962
http://vcg.isti.cnr.it/downloads/downloads.htm

CIGNONI, P., ROCCHINI, C., SCOPIGNO, R. 1998. *Comparison of Mesh Simplification Algorithms*. Computer and Graphics, Volume 22, No. 1. Pages: 37-54. 1998.

CIGNONI, P., ROCCHINI, C., SCOPIGNO, R. 1998. *Metro: Measuring Error on Simplified Surfaces*. Computer Graphics Forum, Volume 17, No. 2. Pages: 167-174. June, 1998.
http://vcg.isti.cnr.it/publications/papers/metro.pdf
http://vcg.isti.cnr.it/downloads/downloads.htm

CLARK, J.H. 1976. *Hierarchical Geometric Models for Visible Surface Algorithms*. Communications of the ACM, Volume 19, Issue 10 (October 1976), Pages: 547-554. 1976.
http://delivery.acm.org/10.1145/370000/360354/p547-clark.pdf

DANAHER, S. 2004. *Digital 3D Design*. Premier Press. 192 pages. September 2004.

DEROSE, T., KASS, M., TRUONG, T. 1998. *Subdivision Surfaces in*

Character Animation. Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques. Pages: 85-94. 1998. http://portal.acm.org/citation.cfm?id=280826

DESCHENES, J.D., HEBERT, P., LAMBERT, P., OUELLET, J.N., TUBIN, D. 2005. *Multiresolution Interactive Modeling with Efficient Visualization.* 5th International Conference on 3-D Digital Imaging and Modeling (3DIM'05). Pages: 39-46. 2005. http://doi.ieeecomputersociety.org/10.1109/3DIM.2005.59

DOO, D., SABIN, M. 1978. *Analysis of the Behaviour of Recursive Division Surfaces near Extraordinary Points.* Computer Aided Design 10, Volume 6. Pages: 356-360, 1978.

DUBE, J.F., BRASSARD, C., VALLET, M.-G., GUIBAULT, F., MAGNAN, R. 2005. *Hydraulic turbine blade reconstruction using subdivision surfaces.* International Conference on Shape Modeling and Applications. Pages: 1-9. Submitted, 2005. http://sympa.polymtl.ca/wws/d_read/dgi-labo-magnu/nos_publications/Geometrie/Construction/SPM2005_subdiv.pdf

DYN, N., LEVIN, D., GREGORY, J.A. 1990. *A butterfly subdivision scheme for surface interpolation with tension control.* ACM Transactions on Graphics, Volume 9, Issue 2. Pages: 160-169. April, 1990. http://portal.acm.org/citation.cfm?id=78956.78958

FINKELSTEIN, A. 2005. *Computer Graphics COS426.* Course Notes (Princeton University). Spring 2005.
http://www.cs.princeton.edu/courses/archive/spr05/cos426/

FLORIANI, L.D., MAGILLO, P. 2002. *Multiresolution Mesh Representation: Models and Data Structures.* Published on the book "Tutorials on Multiresolution in Geometric Modelling". Pages: 363-418. Springer-Verlag. 2002.
http://www.cs.umd.edu/class/fall2003/cmsc828D/primus.pdf

FLORIANI, L.D., MAGILLO, P., MORANDO, F., PUPPO, E. 2000. *Dynamic View-Dependent Multiresolution on a Client-Server Architecture.* Computer-Aided Design, Volume 32, Issue 13. Pages: 805-823, 2000.
http://citeseer.csail.mit.edu/270573.html

FUNKHOUSER, T.A. 2003. *Advanced Topics in Computer Science: Geometric Modeling and Analysis CS597D.* Course Notes (Princeton University). Fall 2003.
http://www.cs.princeton.edu/courses/archive/fall03/cs597D/index.html

GARLAND, M., ZHOU, Y. 2005. *Quadric-based Simplification in any Dimension.* ACM Transactions on Graphics, Volume 24, No. 2. Pages: 209-239. April 2005.
http://portal.acm.org/citation.cfm?id=1061347.1061350

GARLAND, M. 1999. *Quadric-Based Polygonal Surface Simplification.* Ph.D.

thesis, Tech. Rept. CMU-CS-99-105. May, 1999.
http://graphics.cs.uiuc.edu/~garland/research/thesis.html
http://graphics.cs.uiuc.edu/~garland/software/qslim.html

GARLAND, M. 1999. *Multiresolution Modeling: Survey & Future Opportunities*. Eurographics 99 - State of the Art Report. Pages: 111-131. 1999.
http://citeseer.ist.psu.edu/garland99multiresolution.html
http://graphics.cs.uiuc.edu/~garland/papers/STAR99-slides.pdf
http://www.cs.cmu.edu/afs/cs/user/garland/www/multires/survey.html

GARLAND, M., HECKBERT, P.S. 1997. *Surface Simplifying Using Quadric Error Metrics*. Proceeding of SIGGRAPH 1997. Pages: 209-216. 1997.
http://graphics.cs.uiuc.edu/~garland/research/quadrics.html

GOLDMAN, R. 1983. *Two Approaches to a Computer Model for Quadric Surfaces.* IEEE CG&A. Page: 21. September, 1983.

GONSOR, D., NEAMTU, M. 2001. *Surface Subdivision can they be useful for Geometric Modeling Application?* Boeing Technical Report #01-011. December, 2001.
http://www.math.vanderbilt.edu/~neamtu/papers/report.pdf.gz

GOTSMAN, C., GUMHOLD, S., KOBBELT, L.P. 2002. *Simplification and Compression of 3D Meshes*. Published on the book "Tutorials on Multiresolution in Geometric Modelling", Springer-Verlag. Pages: 319-361.

2002.
http://www.cs.technion.ac.il/~gotsman/AmendedPubl/SimplificationAndCompression/SimplificationAndCompression.pdf

GUIBAULT, F. 2003. *INF6800 Conception géométrique assistée par ordinateur et visualisation.* Course Notes (École Polytechnique de Montréal). http://www.cours.polymtl.ca/inf6800/htmlgen/planDeCours.html

HALL, E. 2001. *Efficient Subdivision Surface Evaluation.* A thesis presented to the University of Waterloo. Technical Report Number CS-2000-21. Waterloo, Ontario, Canada. 2001.

HECKBERT, P.S., GARLAND, M. 1999. *Optimal Triangulation and Quadric-Based Surface Simplification.* Journal of Computational Geometry: Theory and Applications, Volume 14, No. 1-3. Pages: 49-65. November, 1999. http://www.cs.cmu.edu/~ph/qtheory.pdf

HECKBERT, P.S., GARLAND, M. 1998. *Simplifying Surfaces with Color and Texture using Quadric Error Metrics.*_Proceedings of the conference on Visualization '98 (IEEE Visualization 98). Research Triangle Park, North Carolina, United States. Pages: 263-269. 1998. http://ieeexplore.ieee.org/iel4/6019/16079/00745312.pdf?arnumber=745312

HECKBERT, P.S., GARLAND, M. 1997. *Survey of Polygonal Surface Simplification Algorithms.* SIGGRAPH 1997, Course Notes: Multiresolution Surface Modeling. 1997.

http://graphics.cs.uiuc.edu/~garland/papers/simp.pdf

HODGINS, J. 2001. *Computer Graphics I 15-462.* Course Notes (Georgia Institute of Technology).
http://www.cs.cmu.edu/afs/cs.cmu.edu/academic/class/15462/web.01f/notes/index.html

HOPPE, H. 1997. *View-Dependent Refinement of Progressive Meshes.* Proceedings of SIGGRAPH 1997. Pages: 189-198. 1997.
http://research.microsoft.com/~hoppe/vdrpm.pdf

HOPPE, H. 1996. *Progressive Meshes.* In ACM SIGGRAPH 1996 Conference Proceedings, Annual Conference Series. Pages: 99-108. August, 1996.
http://research.microsoft.com/~hoppe/pm.pdf

HOPPE, H., DEROSE, T., DUCHAMP, T., HALSTEAD, M., JIN, H., MCDONALD, J., SCHWEITZER, J., STUETZLE, W. 1994. *Piecewise Smooth Surface Reconstruction.* ACM SIGGRAPH. Pages: 295-302. 1994.
http://research.microsoft.com/~hoppe/psrecon.pdf

HOPPE, H., DEROSE, T., DUCHAMP, T., MCDONALD, J., STUETZLE, W. 1993. *Mesh Optimization.* Proceedings of SIGGRAPH 1993. Pages: 19-26. 1993.
http://research.microsoft.com/~hoppe/meshopt.pdf

ISKE, A., QUAK, E., FLOATER, M.S. (Eds.) 2002. *Tutorials on*

Multiresolution in Geometry Modelling. Springer. Series: Mathematics and Visualization. Summer School Lecture Notes. 421 pages. 2002.

JUNKINS, S., HUX, A. 2000. *Subdividing Reality: Employing Subdivision Surfaces for Real Time Scalable 3D*. Game Developers Conference Proceedings. 2000.
ftp://download.intel.com/technology/systems/3d/download/subdivdoc.pdf
ftp://download.intel.com/technology/systems/3d/download/subdiv.pdf
http://www.intel.com/technology/systems/3d/subdiv.htm

KERLOW, I.V. 2004. *The Art of 3D: Computer Animation and Effects*. Third Edition. John Wiley & Sons, Inc., Hoboken, New Jersey. 2004.
http://www.artof3d.com/index.html

KIRCHER, S., GARLAND, M. 2005. *Progressive Multiresolution Meshes for Deforming Surfaces*. Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer animation, Los Angeles, California. Pages: 191-200. 2005.
http://graphics.cs.uiuc.edu/~kircher/adaptive/SCA2005paper171.pdf

KOBBELT, L.P., BISCHOFF, S., KAHLER, K., SCHNEIDER, R., BOTSCH, M., ROSSL, C., VORSATZ, J. 2002. *Geometric Modeling Based on Polygonal Meshes*. MPI Informatik, Saarbrücken. Research Report MPI-I-2000-4-002. 52 pages. July, 2000.
http://domino.mpi-sb.mpg.de/internet/reports.nsf/0/fe74b1173d70be75c12569140044a93b?OpenDocument

KOBBELT, L.P., CAMPAGNA, S., VORSATZ, J., SEIDEL H.-P. 1998. *Interactive Multi-Resolution Modeling on Arbitrary Meshes.* Proceedings of the 25th annual conference on Computer Graphics and Interactive Techniques. Pages: 105-114. 1998.
http://citeseer.ist.psu.edu/kobbelt98interactive.html

LATHROP, O. 1997. *The Way Computer Graphics Works.* Published by John Wiley and Sons. ISBN 0-471-13040-0. 1997.
http://www.dcs.ed.ac.uk/teaching/cs4/www/graphics/Web/intro_graphics/glossary.htm

LEE, A.W.-F. 2000. *Multiresolution Surface Parameterization and Application.* Ph. D. Thesis (Princeton University). November 2000.

LEE, S. 1999. *Interactive Multiresolution Editing of Arbitrary Meshes.* Computer Graphics Forum, Volume 18, Issue 3. September, 1999.
http://www.postech.ac.kr/~leesy/ftp/eg99.pdf

LOOP, C. 1987. *Smooth Subdivision Surfaces Based on Triangles.* M.S. Thesis, Mathematics, University of Utah. 1987.
http://research.microsoft.com/~cloop/thesis.pdf

LOUNSBERY, M., DEROSE, T., WARREN, J. 1997. *Multiresolution Analysis for Surfaces of Arbitrary Topological Type.* ACM Transactions on Graphics, Volume 16, No. 1. Pages: 34-73. January, 1997.

http://portal.acm.org/citation.cfm?id=237750

LUEBKE, D., VARSHNEY, A., COHEN, J., REDDY, M., VARSHNEY, A., WATSON, B., REDDY, M., HUEBNER, R. 2003. GDC (Game Developer Conference -- Level of Detail Management for 3D Games) 2003, Course Materials. 2003.
http://lodbook.com/course/2003/

LUEBKE, D., REDDY, M., COHEN, J.D., VARSHNEY, A., WATSON, B., HUEBNER, R. 2002. *Level of Detail for 3D Graphics.* Elsevier Science Inc. NY, USA. The Morgan Kaufmann Series in Computer Graphics. 2002.

MADI, M., WALTON, D. 2000. *From Hierarchical Structures to Triangular-Loop Structures: A Representation Transformation Algorithm.* Proceedings of the 16th Spring Conference on Computer Graphics 3rd - 6th May, Budmerice, Slovakia. 2000.
http://fractal.dam.fmph.uniba.sk/~sccg/proceedings/2000/Madi.ps.gz

MARTIN, I.M., RONFARD, R., BERNARDINI, F. 2004. *Detail-Preserving Variational Surface Design with Multiresolution Constrains.* International Conference on Shape Modeling and Applications 2004 (SMI'04). Pages: 119-128. 2004.
http://ieeexplore.ieee.org/search/wrapper.jsp?arnumber=1314499

NAYLOR, B., KAUFMAN, A.E., ROSSIGNAC, J., EDELSBRUNNER, H., BAJAJ, C. 1996. *Representations of Geometry for Computer Graphics.*

SIGGRAPH 1996 Course Notes.
http://www.inf.ed.ac.uk/teaching/courses/cg/Web/geometry.pdf

NAYLOR, B.F., KAUFMAN, A.E., ROSSIGNAC, J.R., EDELSBRUNNER, H., POPOVIC, J., HOPPE, H. 1997. *Progressive Simplicial Complexes.* ACM SIGGRAPH 1997. Pages: 217-224. 1997.
http://research.microsoft.com/~hoppe/psc.pdf

PELLACINI, F. 2005. *Computer Graphics CS43.* Course Notes (Dartmouth Computer Science). Fall 2005.
http://www.cs.dartmouth.edu/~cs43/lectures/10_SubdivisionSurfaces_Web.pdf

PIEGL, L., TILLER W. 1997. *The NURBS Book.* Second Edition. Springer-Verlag, New York, NY. 646 pages. 1997.
http://www.csee.usf.edu/~lap/nurbs.htm

PRAUN, E., SWELDENS, W., SCHRODER, P. 2001. *Consistent mesh parameterizations.* Proceedings of ACM SIGGRAPH 01. Pages: 179-184. August, 2001.
http://www.multires.caltech.edu/pubs/consistent.pdf

ROY, M., FOUFOU, S., TRUCHETET, F. 2004. *Mesh Comparison Using Attribute Deviation Metric.* International Journal of Image and Graphics (IJIG), Volume 4, No. 1. Pages: 127-140. January, 2004.
http://meshdev.sourceforge.net/files/Roy-Ijig2004.pdf

http://meshdev.sourceforge.net/

ROY, M., FOUFOU, S., TRUCHETET, F. 2002. *Generic Attribute Deviation Metric for Assessing Mesh Simplification Algorithm Quality.* Proceedings of IEEE International Conference on Image Processing. Rochester, USA. Pages: 817-820. September, 2002.
http://imaging.utk.edu/people/former/mroy/files/Roy-Icip02.pdf
http://imaging.utk.edu/people/former/mroy/files/Roy-Icip02-Poster.pdf

ROY, M., NICOLIER, F., FOUFOU, S., TRUCHETE, F., KOSCHAN, A., ABIDI, M. 2002. *Assessment of Mesh Simplification Algorithm Quality.* Proceedings of SPIE Electronic Imaging, Vol. 4661, Pages: 128-137, San Jose, USA, January 2002.
http://imaging.utk.edu/people/former/mroy/files/Roy-Ei02.pdf
http://imaging.utk.edu/people/former/mroy/files/Roy-Ei02.ppt

SARFRAZ, M. 2004. *Advances in Geometric Modeling.* John Wiley. 334 pages. March 2004.

SCHRODER, P. 2004. *Subdivision: Smooth Bases on Meshes (and Other Neat Tricks).* Oberwolfach Seminar on Discrete Differential Geometry. Spring 2004.
http://www.multires.caltech.edu/Oberwolfach2004/SubdivisionWeb.pdf

SCHRODER, P. 2001. *Subdivision, Multiresolution and the Construction of Scalable Algorithms in Computer Graphics.* Multivariate Approximation and

Applications. Cambridge University Press. 2001.
http://www.multires.caltech.edu/pubs/eilat.pdf

SCHRODER, P. 1999. *Opportunities for Subdivision-Based Multiresolution Modeling*. Proceedings of the 7th Pacific Conference on Computer Graphics and Applications. Pages: 104-105. Oct, 1999.
http://doi.ieeecomputersociety.org/10.1109/PCCGA.1999.803353

SCHROEDER, P. 1997. *A Topology-Modifying Progressive Decimation Algorithm*. 8th Proceedings of IEEE Visualization 1997 (Vis'97). Pages: 205-212. 1997.
http://doi.ieeecomputersociety.org/10.1109/VISUAL.1997.663883

SHAFFER, E., GARLAND, M. 2005. *A Multiresolution Representation for Massive Meshes*. IEEE Transactions on Visualization and Computer Graphics, Volume 11, Issue 2. Pages: 139-148. March, 2005.
http://graphics.cs.uiuc.edu/~shaffer1/papers/mm-mesh.pdf

SHARP, B. 2000. *Subdivision Surface Theory*. Gamasutra. April, 2000.
http://www.gamasutra.com/features/20000411/sharp_01.htm

SILVA, F.G.M., GOMES, A.J.P. 2004. *Interactive Editing of Multiresolution Meshes*. Computer Graphics and Image Processing, XVII Brazilian Symposium on (SIBGRAPI'04). Pages: 202-209. October, 2004.
http://doi.ieeecomputersociety.org/10.1109/SIBGRA.2004.1352962

SILVA, F.G.M., GOMES, A.J.P. 2003. *Adjacency and Incidence Framework – a Data Structure for Efficient and Fast Management of Multiresolution Meshes.* ACM (Association for Computing Machinery). 2003. http://portal.acm.org/ft_gateway.cfm?id=604503&type=pdf

STAM, J., LOOP C. 2003. *Quad/Triangle Subdivision.* Computer Graphics Forum, Volume 22, No. 1. Pages: 79-85(7). March, 2003. http://research.microsoft.com/~cloop/thesis.pdf

SURAZHSKY, V., GOTSMAN, C. 2005. *A Qualitative Comparison of Some Mesh Simplification Software Packages.* Preprint. March 2005. http://www.cs.technion.ac.il/~gotsman/AmendedPubl/Vitaly/SimpStudy-final.pdf

SUZUKI, H., SAKURAI, Y., KANAI, T., KIMURA, F. 1998. *Interactive mesh dragging with adaptive remeshing technique.* Computer Graphics and Applications, Pacific Graphics 98, Sixth Pacific Conference on 26-29 Oct. 1998. Pages: 188-197. 1998. http://ieeexplore.ieee.org/search/wrapper.jsp?arnumber=732112

VINCE, J., EARNSHAW, R. (Eds.) 2002. *Advances In Modelling, Animation, And Rendering.* Springer-Verlag UK. Proceedings of Computer Graphics International 2002. 546 pages. July, 2002.

WARREN, J., SCHAEFER, S. 2004. *A Factored Approach to Subdivision Surfaces.* IEEE Computer Graphics and Applications, Volume 24, No. 3.

Pages: 74-81. May/June 2004.
http://doi.ieeecomputersociety.org/10.1109/MCG.2004.1297015

WARREN, J., WEIMER, H. 2002. *Subdivision Methods for Geometric Design: A Constructive* Approach. The Morgan Kaufmann Series in Computer Graphics. 2002.
http://www.subdivision.org/subdivision/book/book.jsp

WARREN, J. 1994. *Subdivision Methods for Geometric Design.* Unpublished manuscript that forms an initial draft of the book "Subdivision Methods for Geometric Design: A Constructive Approach". 1994.
http://www.cs.rice.edu/~jwarren/papers/book.ps.gz

WU, Y., He, Y., CAI, H. 2004. *QEM-based mesh simplification with global geometry features.* Proceedings of the 2nd international conference on Computer Graphics and Interactive Techniques in Australasia and South East Asia, Singapore. Pages: 50-57. 2004.
http://portal.acm.org/citation.cfm?id=988843

ZORIN, D.N., SCHRODER, P., DEROSE, T., KOBBELT, L.P., LEVIN, A., SWELDENS, W. 2000. *Subdivision for animation and modeling.* Course Notes, SIGGRAPH 2000.
http://mrl.nyu.edu/publications/subdiv-course2000/coursenotes00.pdf

ZORIN, D.N. 1998. *Stationary Subdivision and Multiresolution Representation.* California Institute of Technology. Doctoral Thesis. January

1998.
http://caltechcstr.library.caltech.edu/180/01/postscript.pdf

ZORIN, D.N., HOLST, M., SCHRODER, P. 1997. *Subdivision-Based Surface Representation.* Atlanta, GA. TeamCAD 97 Workshop. Pages: 35-38. 1997.
http://www.scicomp.ucsd.edu/~mholst/pubs/dist/ZHS97a.pdf

ZORIN, D.N., SCHRODER, P., SWELDENS, W. 1997. *Interactive Multiresolution Mesh Editing.* Proceedings of the 24th annual conference on Computer Graphics and Interactive Techniques. Pages: 259-268. 1997.
http://graphics.stanford.edu/~dzorin/multires/meshed/index.html

ZORIN, D.N., SCHRODER, P., SWELDENS, W. 1996. *Interpolating Subdivision for meshes with arbitrary topology.* Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques. Pages: 189-192. 1996.
http://portal.acm.org/citation.cfm?id=237254

ProgMesh: A Windows application for 3D polygon mesh simplification which supports the 3D Studio file format. It is based on the progressive mesh (H. HOPPE).
http://www.paralelo.com.br/produtos/progmesh/progmesh.htm

Princeton Shape Retrieval and Analysis Group. *3D Model Search Engine.*
http://shape.cs.princeton.edu/search.html

APPENDIX I

Related Statistics from Evaluation

Due to space limitations, in this appendix we supply some tables including statistics as the reference of related chapters.

Table I-1 lists the concrete deviation rates of performing Loop subdivision on the cup surfaces from level 1 to level 6. For performing Modified Butterfly subdivision on the same initial surface, the concrete values are listed in Table I-2. The values in Table I-1 come from Table 4-3 and the values in Table I-2 come from Table 4-4.

Table I -1: Deviation rates of performing Loop subdivision on the cup surfaces from level 1 to level 6.

	Level 1	Level 2	Level 3	Level 4	Level 5	Level 6
$\triangle c = \sigma_c / \bar{c}$	0.0405	0.0233	0.0121	0.0114	0.0182	0.0105
$\triangle n = \sigma_n / \bar{n}$	0.0610	0.0109	0.0122	0.0196	0.0102	0.0045
$\triangle s = \sigma_s / \bar{s}$	0.1750	0.0436	0.0410	0.0148	0.0028	0.0014

Table I-3 lists the concrete deviation rates of performing Loop and Modified Butterfly subdivision on the torus surfaces from level 1 to level 3. Similarly, Table I-4 lists the concrete deviation rates from the same evaluation on the pawn surfaces. Table I-3/ Table I-4 respectively reference Table 4-6/Table 4-7.

Table I -2: Deviation rates of performing Modified Butterfly subdivision on the cup surfaces from level 1 to level 6.

	Level 1	Level 2	Level 3	Level 4	Level 5	Level 6
$\Delta c = \sigma_c / \bar{c}$	0.0386	0.0200	0.0136	0.0066	0.0104	0.0125
$\Delta n = \sigma_n / \bar{n}$	0.0424	0.0215	0.0067	0.0127	0.0093	0.0053
$\Delta s = \sigma_s / \bar{s}$	0.0616	0.0251	0.0294	0.0144	0.0122	0.0485

Table I -3: Deviation rates of performing Loop and Modified Butterfly subdivision on the torus surfaces from level 1 to level 3.

	Loop Subdivision			Modified Butterfly Subdivision		
	Level 1	Level 2	Level 3	Level 1	Level 2	Level 3
$\Delta c = \sigma_c / \bar{c}$	0.0316	0.0399	0.0158	0.1153	0.0261	0.0232
$\Delta n = \sigma_n / \bar{n}$	0	0.047	0.0034	0.0399	0.0379	0.0323
$\Delta s = \sigma_s / \bar{s}$	0.0437	0.0258	0.0397	0.0730	0.0371	0.1174

Table I -4: Deviation rates of performing Loop and Modified Butterfly subdivision on the pawn surfaces from level 1 to level 3.

	Loop Subdivision			Modified Butterfly Subdivision		
	Level 1	Level 2	Level 3	Level 1	Level 2	Level 3
$\Delta c = \sigma_c / \bar{c}$	0.0507	0.0111	0.0092	0.0453	0.0141	0.0075
$\Delta n = \sigma_n / \bar{n}$	0.0511	0.0368	0.0307	0.0081	0.0324	0.0213
$\Delta s = \sigma_s / \bar{s}$	0.0266	0.0525	0.0116	0.1036	0.0128	0.0116

In order to compare the difference between with local support and without local support, we list relevant statistics on cup, torus and pawn surface separately in Table I-5, Table I-6 and Table I-7. We once illustrated them respectively in Figure 4-7, Figure 4-8 and Figure 4-9. The differences between the total costs with local support and without local support are listed in Table I-8, which is illustrated in Figure 4-10.

Table I -5: The comparison between with local support and without local support on the cup surfaces.

				Perform Subdivision	Mesh Connectivity	Surface Normals	Total Time Costs
Cup	Local Support	Case 1	Level 1	0.0001	0.0000	0.0002	0.0003
			Level 2	0.0016	0.0000	0.0007	0.0023
			Level 3	0.0170	0.0000	0.0029	0.0199
		Case 2	Level 1	0.0002	0.0000	0.0002	0.0004
			Level 2	0.0037	0.0000	0.0007	0.0044
			Level 3	0.0219	0.0000	0.0027	0.0246
	Full Compute	Case 1	Level 1	0.0011	0.0080	0.0002	0.0093
			Level 2	0.0050	0.0289	0.0007	0.0346
			Level 3	0.0187	0.1057	0.0026	0.127
		Case 2	Level 1	0.0012	0.0081	0.0002	0.0095
			Level 2	0.0035	0.0307	0.0006	0.0348
			Level 3	0.0176	0.1082	0.0026	0.1284

Table I -6: The comparison between with local support and without local support on the torus surfaces.

				Perform Subdivision	Mesh Connectivity	Surface Normals	Total Time Costs
Torus	Local Support	Case 1	Level 1	0.0002	0.0000	0.0001	0.0003
			Level 2	0.0032	0.0000	0.0003	0.0035
			Level 3	0.0183	0.0000	0.0010	0.0193
		Case 2	Level 1	0.0002	0.0000	0.0001	0.0003
			Level 2	0.0036	0.0000	0.0003	0.0039
			Level 3	0.0192	0.0000	0.0010	0.0202
	Full Compute	Case 1	Level 1	0.0003	0.0025	0.0001	0.0029
			Level 2	0.0015	0.0124	0.0003	0.0142
			Level 3	0.0133	0.0446	0.0011	0.059
		Case 2	Level 1	0.0002	0.0021	0.0001	0.0024
			Level 2	0.0011	0.0118	0.0003	0.0132
			Level 3	0.0051	0.0471	0.0010	0.0532

Table I -7: The comparison between with local support and without local support on the pawn surfaces.

				Perform Subdivision	Mesh Connectivity	Surface Normals	Total Time Costs
Pawn	Local Support	Case 1	Level 1	0.0001	0.0000	0.0008	0.0009
			Level 2	0.0047	0.0000	0.0028	0.0075
			Level 3	0.0232	0.0000	0.0114	0.0346
		Case 2	Level 1	0.0001	0.0000	0.0007	0.0008
			Level 2	0.0047	0.0000	0.0028	0.0075
			Level 3	0.0244	0.0000	0.0117	0.0361
	Full Compute	Case 1	Level 1	0.0032	0.0291	0.0006	0.0329
			Level 2	0.0207	0.1011	0.0025	0.1243
			Level 3	0.0510	0.3979	0.0113	0.4602
		Case 2	Level 1	0.0064	0.0275	0.0006	0.0345
			Level 2	0.0130	0.1081	0.0025	0.1236
			Level 3	0.0486	0.4099	0.0114	0.4699

Table I -8: The differences between total time costs.

		Level 1	Level 2	Level 3
Cup	Case 1	0.009	0.0323	0.1071
	Case 2	0.0091	0.0304	0.1038
Torus	Case 1	0.0026	0.0107	0.0397
	Case 2	0.0021	0.0093	0.033
Pawn	Case 1	0.032	0.1168	0.4256
	Case 2	0.0337	0.1161	0.4338

APPENDIX II

Introduction of MSSE System

1. Compiling Environment

The MSSE (Multi-resolution Subdivision Surface Editing) system is developed under Microsoft Visual Studio .NET 2003 installed with OpenGL. The release can be executed in Windows environment.

2. Source Data & File Format

Source data, as the input of multi-resolution surface meshes modeling systems, describe the models' geometry and topology. They influence the final results of models particularly. Source data can contain different information to describe the same models. In Figure II-1, we illustrate that Loop subdivision passes through two cube models with the "same" cube surface but containing different source data. From this illustration, we can observe that source data play an important role in influencing model appearance.

Most of time, source data are composed of lists of numbers illustrating vertices' positions, edges (connections between vertices) and faces (connections among edges), which are viewed as numerical descriptions (see Figure II- 2) of surface meshes. Currently, there are various file formats to transport models' numerical descriptions. Some common file formats are listed as following: *.obj, *.3dm, *.dxf, *mel, *.stl, *.wrl, and *.pie.

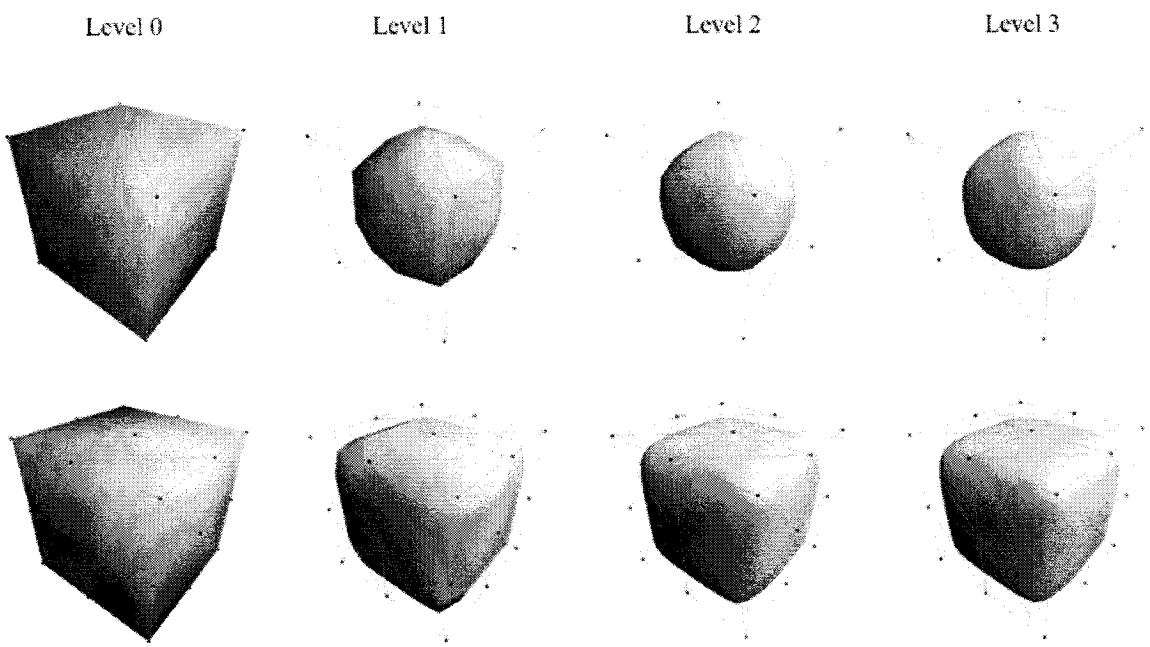


Figure II -1: The illustration of Loop subdivision passing through two cube surfaces containing different source data.

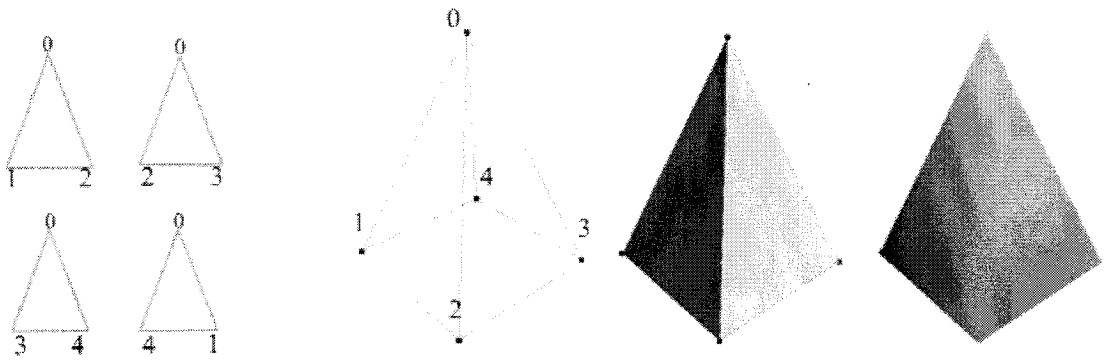
A few modeling software supply conversions among common file formats. In our MSSE system, most of models used in our project are collected from the link, whose file format is .ray:
http://www.cs.princeton.edu/courses/archive/spr04/cos426/assn3/

Later we collected some models from 3dcafe, whose link is:
www.3dcafe.com

And one horse model is collected from:
http://www.cs.ucsb.edu/~cs280/winter2004/hw3/

A few collected models are with .obj file format. After analyzing the file

formats of existing models, we wrote some codes to convert them into *.ray. Later, in order to read source files more directly and more conveniently, we implement these conversions into MSSE system. Our editing system is capable to read, modify and save source model files with the formats: *.obj, *.ray and *.pie directly.



File format #1

Point	X	Y	Z				
0	-1	0	-1	Face	Point	Point	Point
1	0	-1	-1	Face	4	3	2
2	0	0	1	Face	3	0	2
3	0	1	-1	Face	0	1	2
4	1	0	-1	Face	1	4	2

File format #2

V	-1	0	-1	
V	0	-1	-1	f 5//5 4//4 3//3
V	0	0	1	f 4//4 1//1 3//3
V	0	1	-1	f 1//1 2//2 3//3
V	1	0	-1	f 2//2 5//5 3//3

Figure II -2: The same model cone with two different file formats.

The cube model with *.obj* format is simply shown as bellows:

Rhino
g object_1
/* Vertex x y z*/
v -1 -1 -1
v -1 -1 0
v -1 -1 1
v -1 0 -1
...

/* Vertex Normal x y z*/
vn 0.8164966106414795 0.4082483053207398 0.4082483053207398
vn 0.7071067690849304 0.7071067690849304 0
vn 0.3333333432674408 0.6666666865348816 -0.6666666865348816
...

/*Face vextex1// vextex1, vertex 2//vextex2, vertex 3//vertex3*/
f 18//18 21//21 13//13
f 18//18 13//13 10//10
f 21//21 24//24 15//15
f 21//21 15//15 13//13
...

The dialog of opening one file is shown in Figure II- 3:

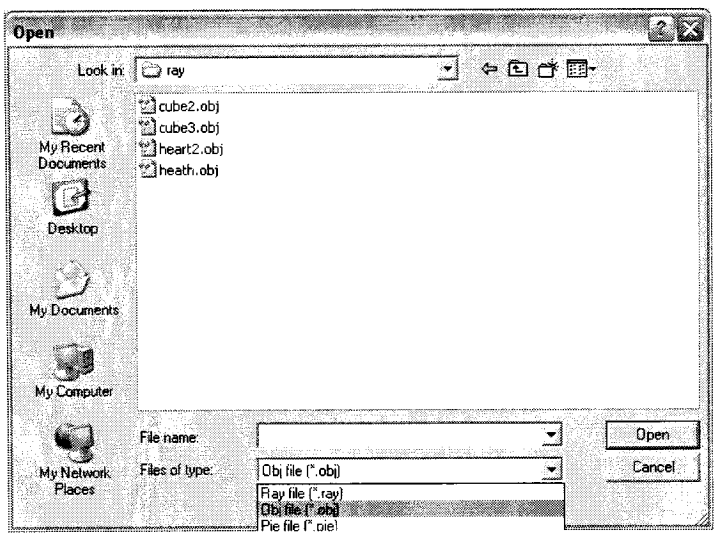


Figure II -3: The dialog used to open one file with the format .obj.

In our program, we set a filter in the open file dialog, for which we use one function *lpstrFilter( )*. The relevant codes can be referenced as bellows:

```
BOOL shortcutfilename(int num, char* filename)
{
    OPENFILENAME ofn;
    memset(&ofn, 0, sizeof(OPENFILENAME));
    ofn.lStructSize = sizeof(OPENFILENAME);
    ofn.lpstrFile = filename;
    ofn.nMaxFile = 512;
    if ( num == 0)
    {
        ofn.lpstrFilter = "Ray file (*.ray)\0*.ray\0Obj file (*.obj)\0*.obj\0Pie file (*.pie)\0\0";
        return(GetOpenFileName( &ofn));
    }
    else if (num == 1)
    {
        ofn.lpstrFilter = "Ray file (*.ray)\0*.ray\0\0";
        return(GetSaveFileName(&ofn));
    }
    return -1;
}
```

3. Graphical User Interface

The graphical user interface composes two parts (see Figure II-4): the part on the left is to display surfaces and the part on the right is a menu used for editing. The menu includes six zones: (1). Subdivision zone is used to select subdivision level and control mesh level. The icon "Define Crease" in subdivision zone is used to define crease on Loop or Modified Butterfly subdivision surfaces. For Loop subdivision, if each time we only select one vertex to define crease, the editing system will automatically perform vertex crease (see Figure II- 5 and Figure II- 6). Otherwise, each time when we selected two vertices to define crease, the editing system will perform edge crease (see Figure II- 7). (2). Display zone is used to show control mesh or

crease etc. For example, when we define crease, we can choose to display or not to display the selected edges before they are defined as creases (see Figure II- 8). (3). Shading zone is used to display surface in different modes (see Figure II- 9). (4). Drag Mode zone is used to move the selected vertex on different way. (5). Subdivision Mode zone is used to choose Loop subdivision or Modified Butterfly subdivision. (6). File zone is used to open or save control or subdivision meshes.

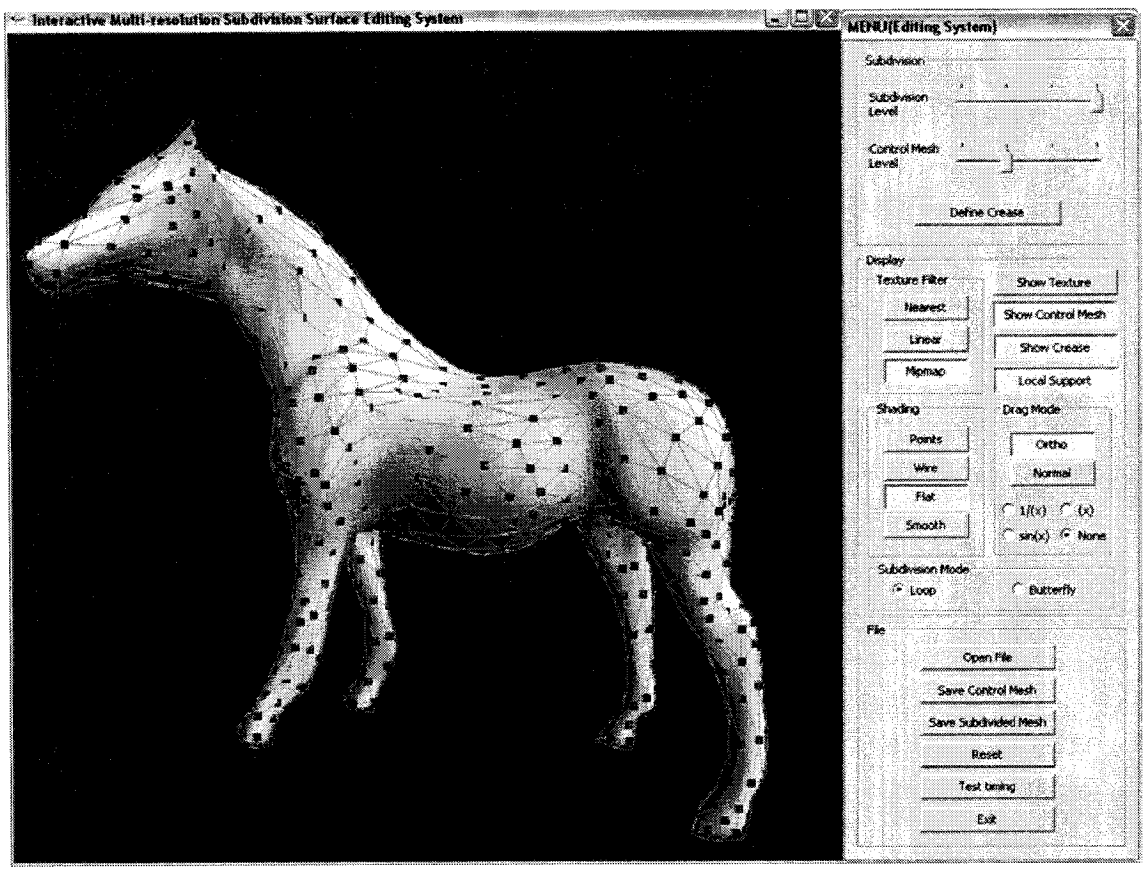


Figure II -4: The graphical user interface.

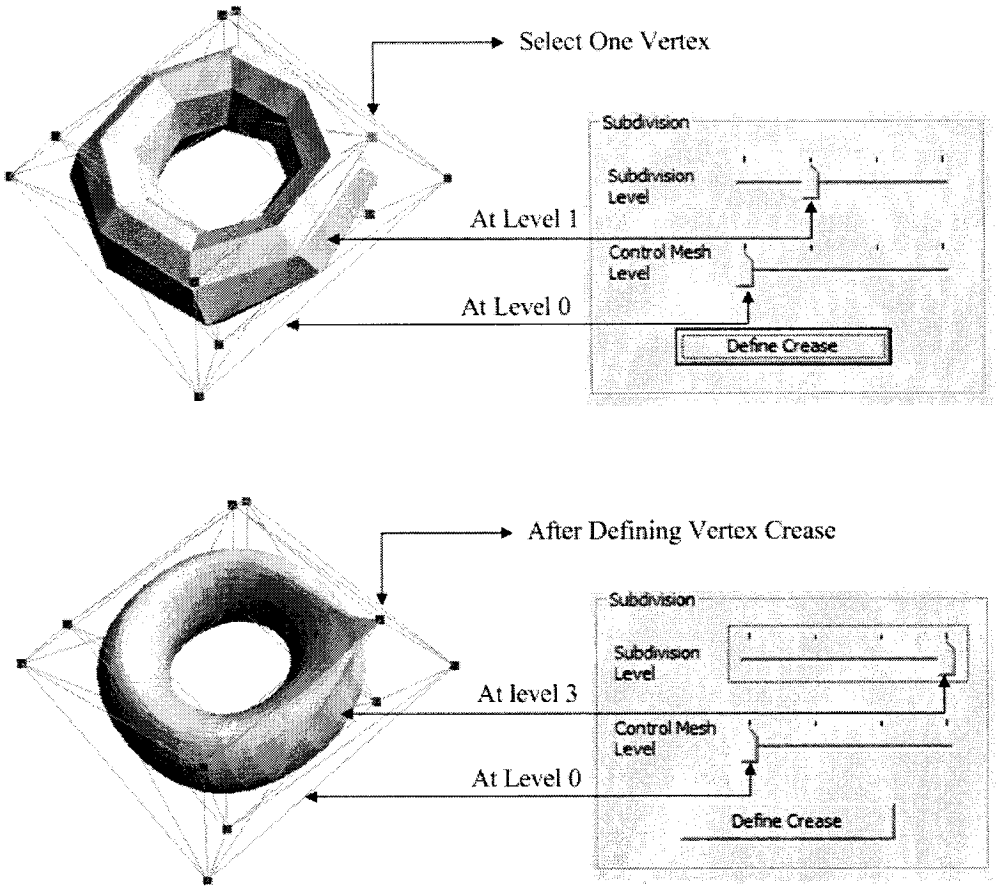


Figure II -5: Define vertex crease on Loop subdivision-based torus surface.

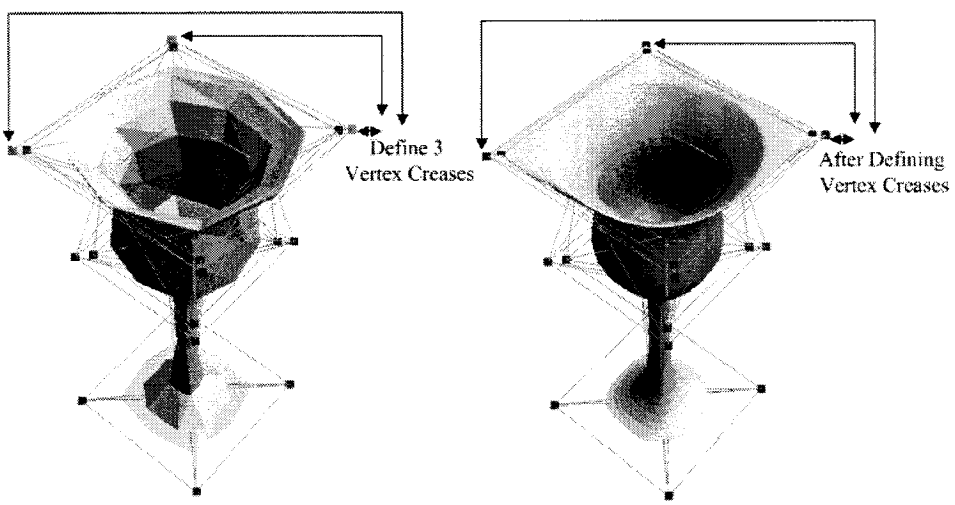


Figure II -6: Define 3 vertex creases on Loop subdivision-based cup surface.

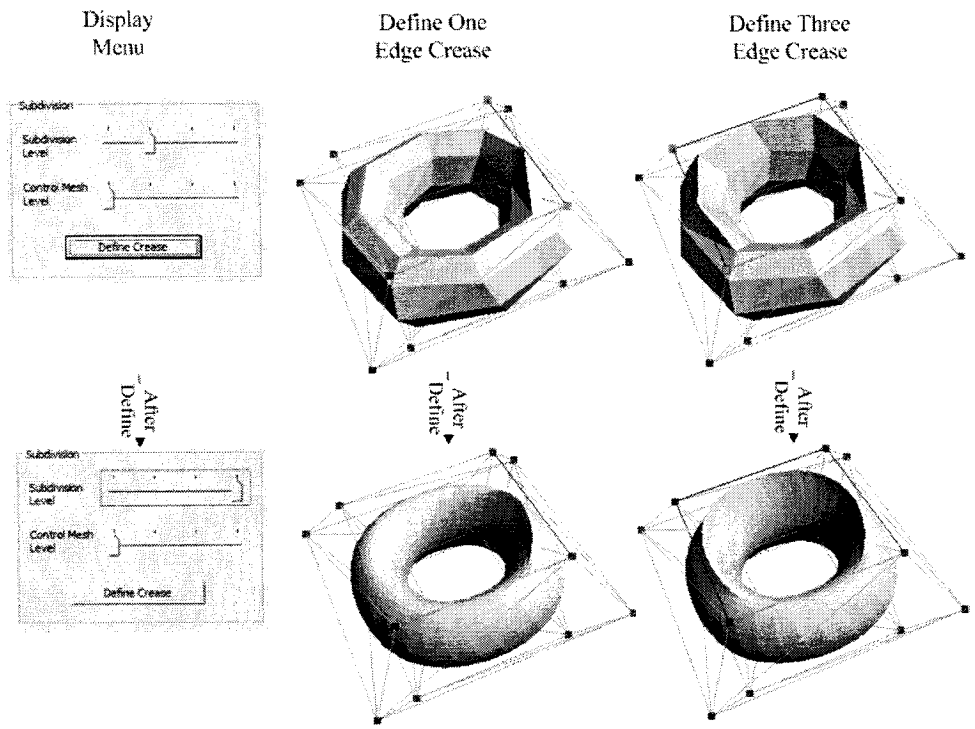


Figure II -7: Define 3 edge creases on the Loop subdivision torus surface.

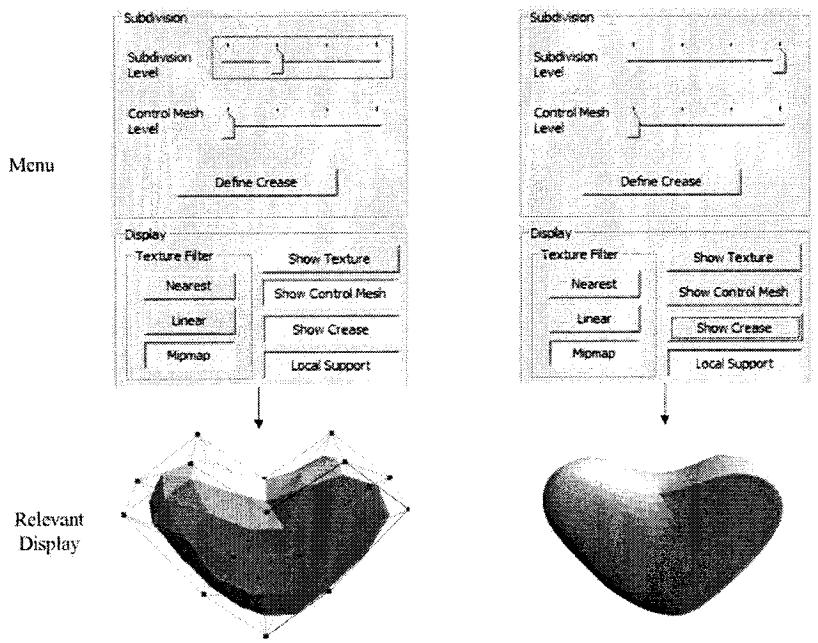


Figure II -8: Define edge creases on the Loop subdivision heart surface.

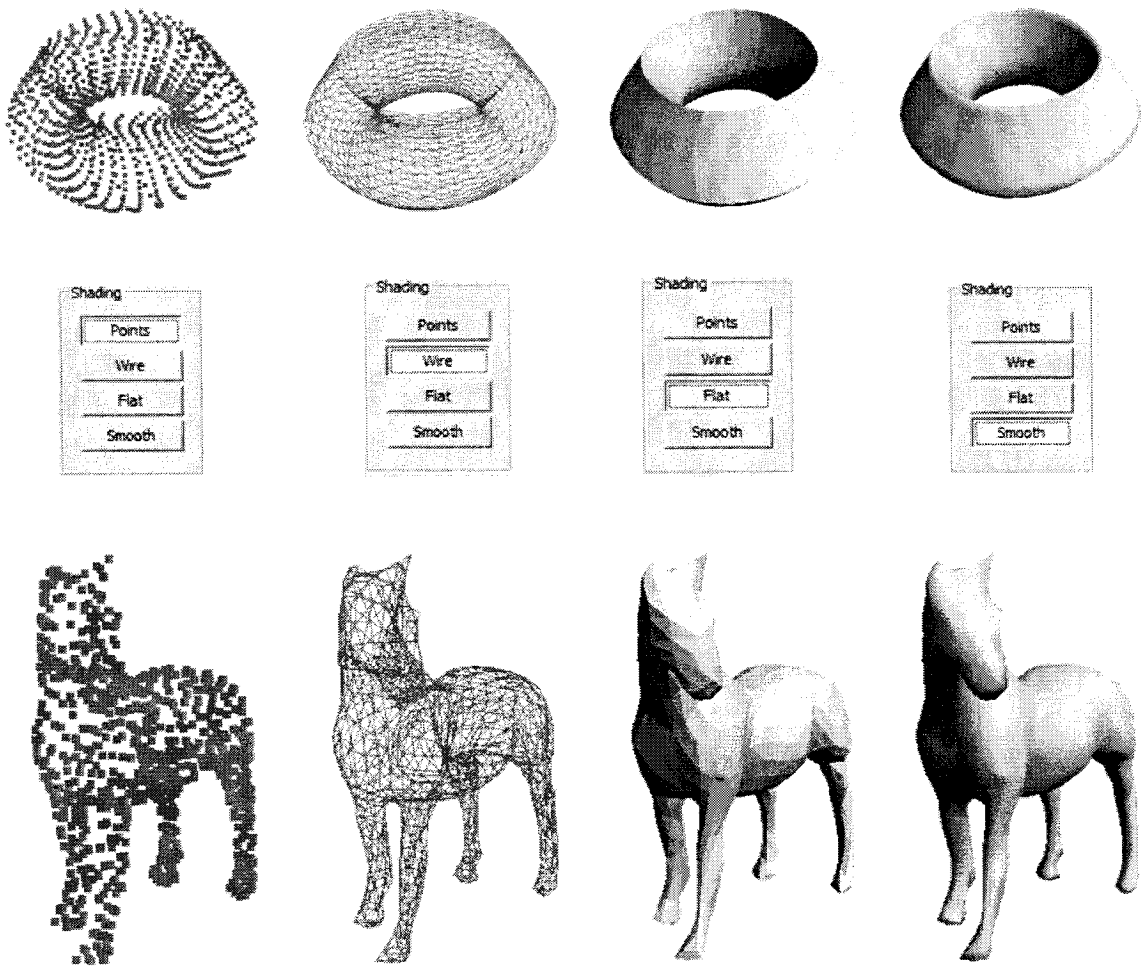


Figure II -9: Illustration of different shading modes.

In the menu, two icons "Test Timing" and "Local Support" used to evaluate system performance are added also. The icon "Test Timing" tests time costs of performing subdivision, computing mesh connectivity and surface normals on current test surface (see Figure II-10) including the statistics of standard deviation values (see Figure II-11). The icon "Local Support" is to switch between the function with local support and without local support (see Figure II- 12).

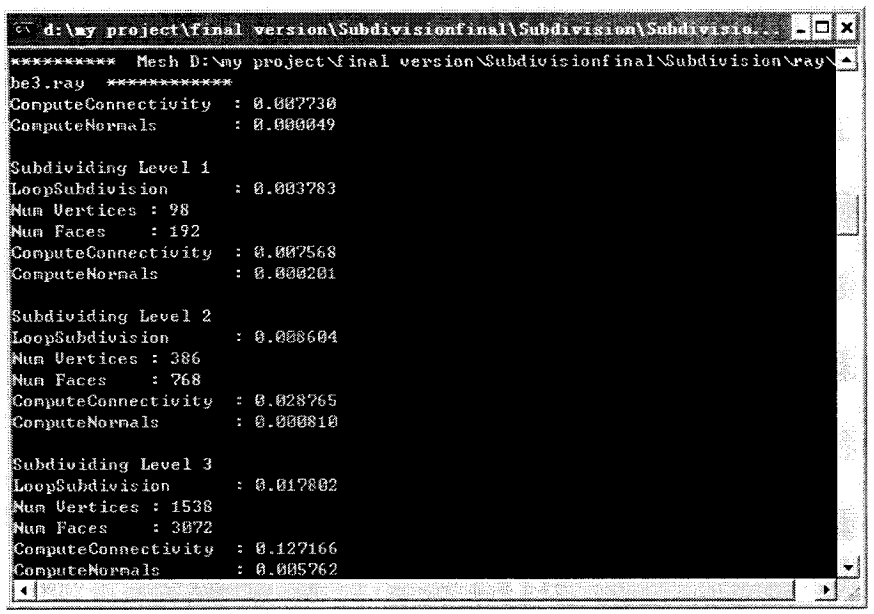


Figure II -10: Illustration of performing "testing timing".

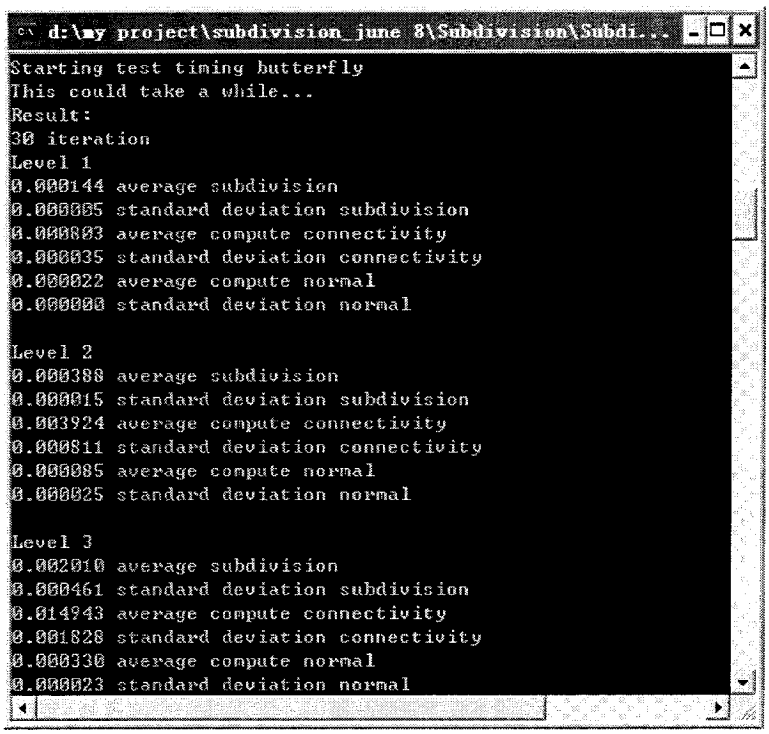


Figure II -11: Illustration of achieving the statistics of standard deviation values by using "testing timing".

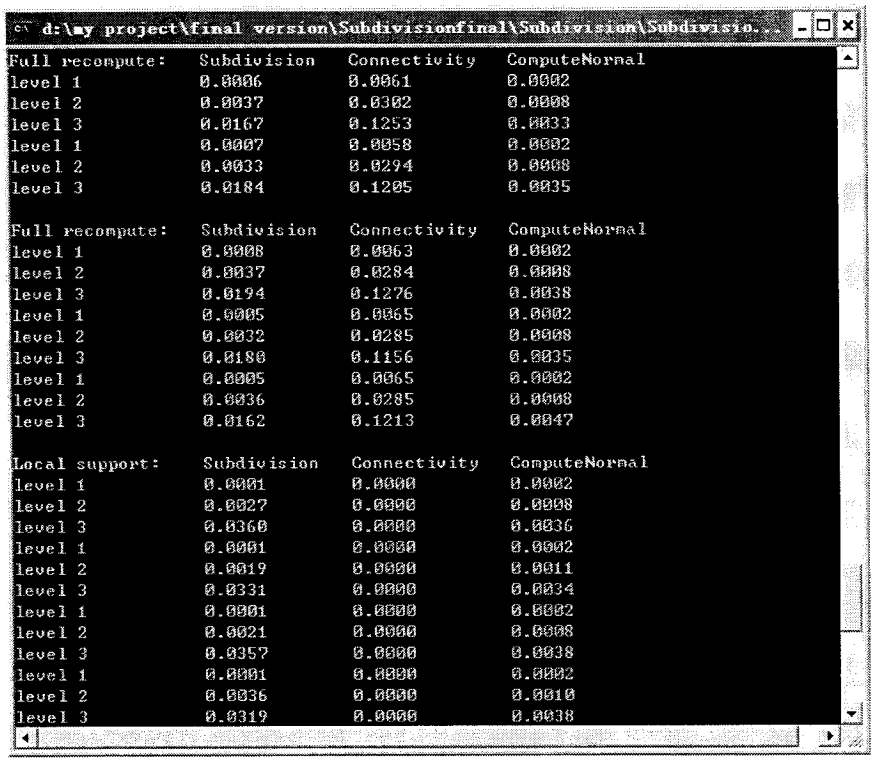


Figure II -12: Illustration of verifying local support.

APPENDIX III

Related Software and Useful Tools

1. Software for Modeling:

(1). Rhinoceros ®– NURBS Modeling for Windows:

http://www.rhino3d.com/

In our project, a few of models as input files are created by using Rhinoceros. Rhinoceros provides a nice designing environment in Windows for creating and analyzing surface meshes accurately. Rhinoceros holds the strong compatibility for illustrating and rendering models, which helps us convert some models with different formats that were collected from Internet or other places into files with the format .obj. Meanwhile, Rhinoceros is a kind of practical tool with a favourable performance. Even though on the Laptop with only 256MB memory, rendering a huge surface mesh is adequately smooth. Moreover, Rhinoceros is very useful for modeling NURBS. Personally, we once used Rhinoceros to create raceways of motorcycle for the project running in the CAVE (Cave Automatic Virtual Environment), which achieved nice results.

(2). Maya ©Alias:

http://www.alias.com/eng/index.shtml

Maya is very powerful for animation. In comparison with Rhinoceros, possibly Maya is better in applying textures and lights. The personal opinion is that Maya possesses much more memory and it is pretty heavy to do

modeling on ordinary laptops.

(3). 3D Studio Max © Discreet

http://www.idigitalemotion.com/tutorials/studiomax.html

(4). ZBrush © Pixologic

http://pixologic.com/home/home.shtml

2. Related Software

(1). The basic structure of MSSE system references the assignment in the following link:

http://www.cs.princeton.edu/courses/archive/spr04/cos426/assn3/

(2). Subdivision 2.0

This software implements Loop subdivision and Catmull-Clark subdivision. The input file is with .vrml format.

http://mrl.nyu.edu/~biermann/subdivision/

3. Related Tools & Links:

(1). Simplification Software

Jade (Just Another DEcimator) is based on a global error measurement. It is developed by the Visual Computer Group of CNUCE/IEI - C.N.R. at Pisa.

http://vcg.isti.cnr.it/activities/surfacegrevis/simplification/jade_html/jadeinfo.html

QSlim is based on quadric-based simplification algorithm which was developed by Michael Garland.

http://graphics.cs.uiuc.edu/~garland/software/qslim.html

(2). Software Used to Compare the Differences between Surfaces

M.E.S.H. (Measuring Error between Surfaces using the Hausdorff distance) is used to measure distortion between two given triangular meshes. M.E.S.H. is based on the Hausdorff distance to calculate maximum, mean, and root-mean-square errors between surfaces. The software can read input models with .ply format.

The related link is:

http://mesh.berlios.de/

The related paper is:

http://mesh.berlios.de/mesh.pdf

MeshDev uses attribute Deviation metric to assess mesh simplification algorithm quality.

http://meshdev.sourceforge.net/publications.html

Metro can be used to evaluate surface difference by returning both numerical results and visual results.

http://vcg.iei.pi.cnr.it/metro.html

(3). Source Models

Turbine Blade:

http://www.cc.gatech.edu/data_files/large_models/blade.ply.gz

Stanford Bunny:

ftp://graphics.stanford.edu/pub/3Dscanrep/bunny.tar.gz

http://www.cc.gatech.edu/data_files/large_models/bunny.ply.gz

(4). Useful Library

The link of GLOD (Geometric Level of Detail) library is:

http://www.cs.jhu.edu/~graphics/GLOD/

APPENDIX IV

Photo Gallery of Results

1. Loop Subdivision & Modified Butterfly Subdivision Surfaces

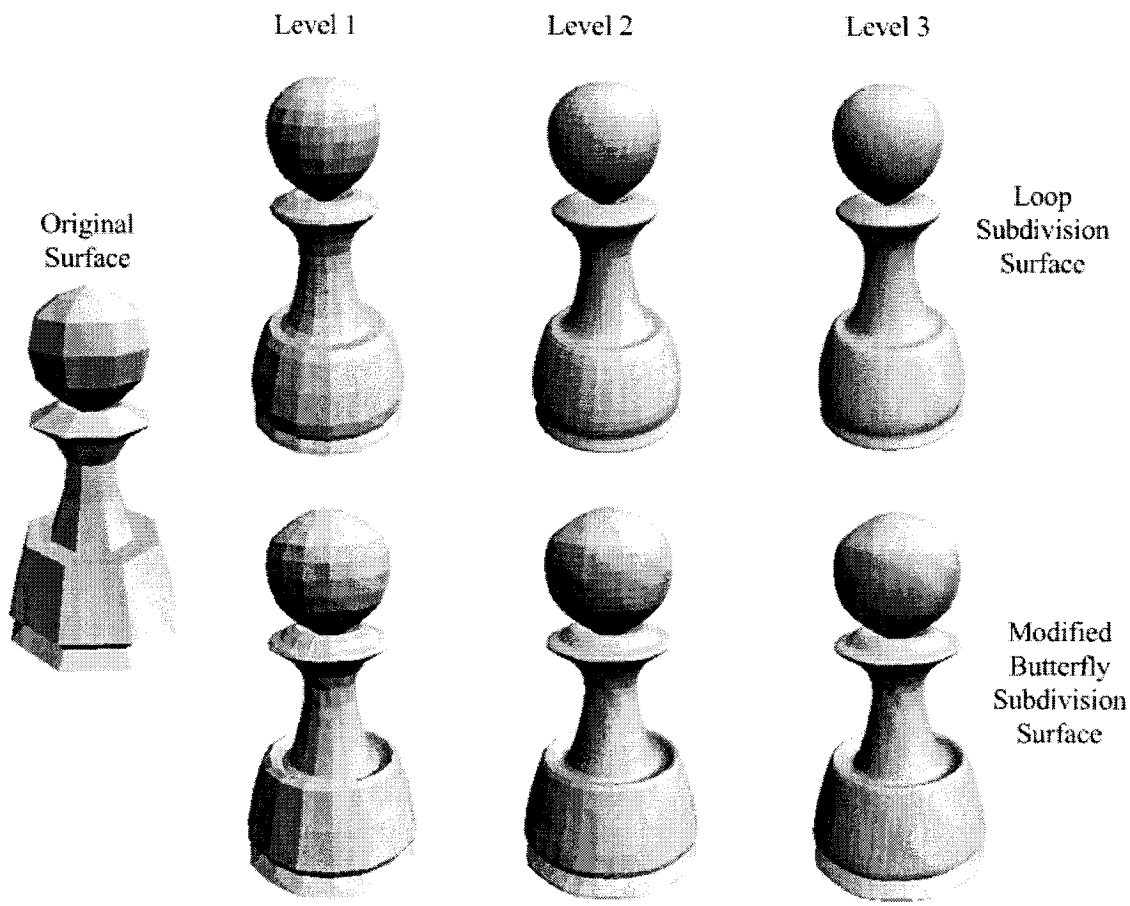


Figure IV -1: The Loop or Modified Butterfly subdivision-based pawn surfaces.

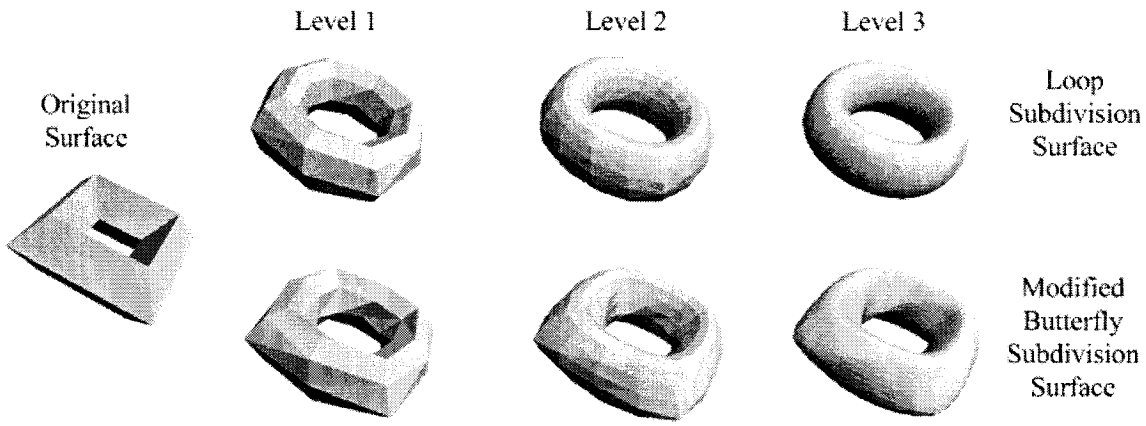


Figure IV -2: The Loop/Modified Butterfly subdivision-based torus surfaces.

2. Creating Creases on Subdivision Surfaces

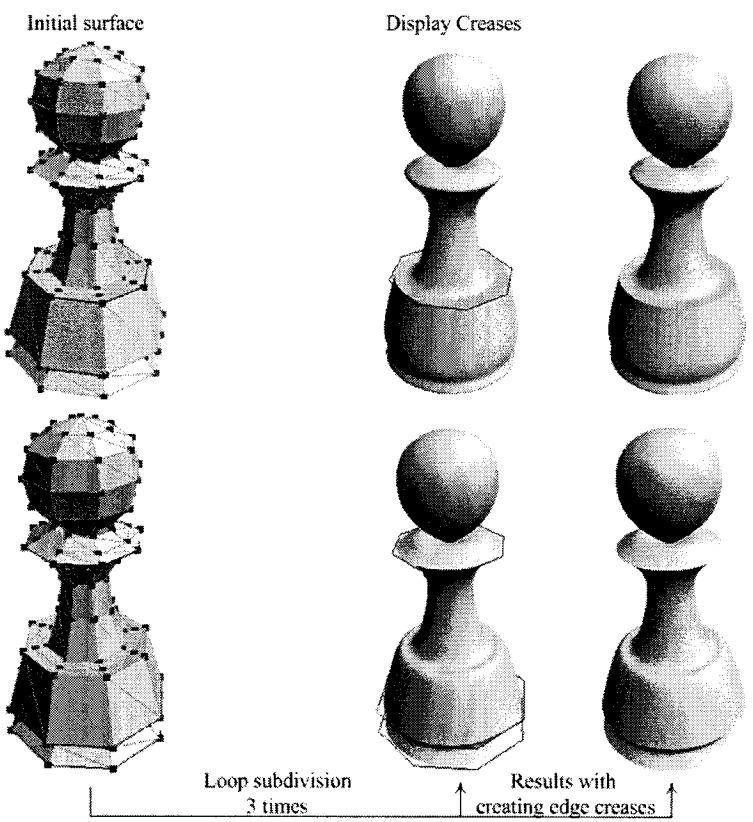


Figure IV -3: Create edge creases on the Loop subdivision pawn surfaces.

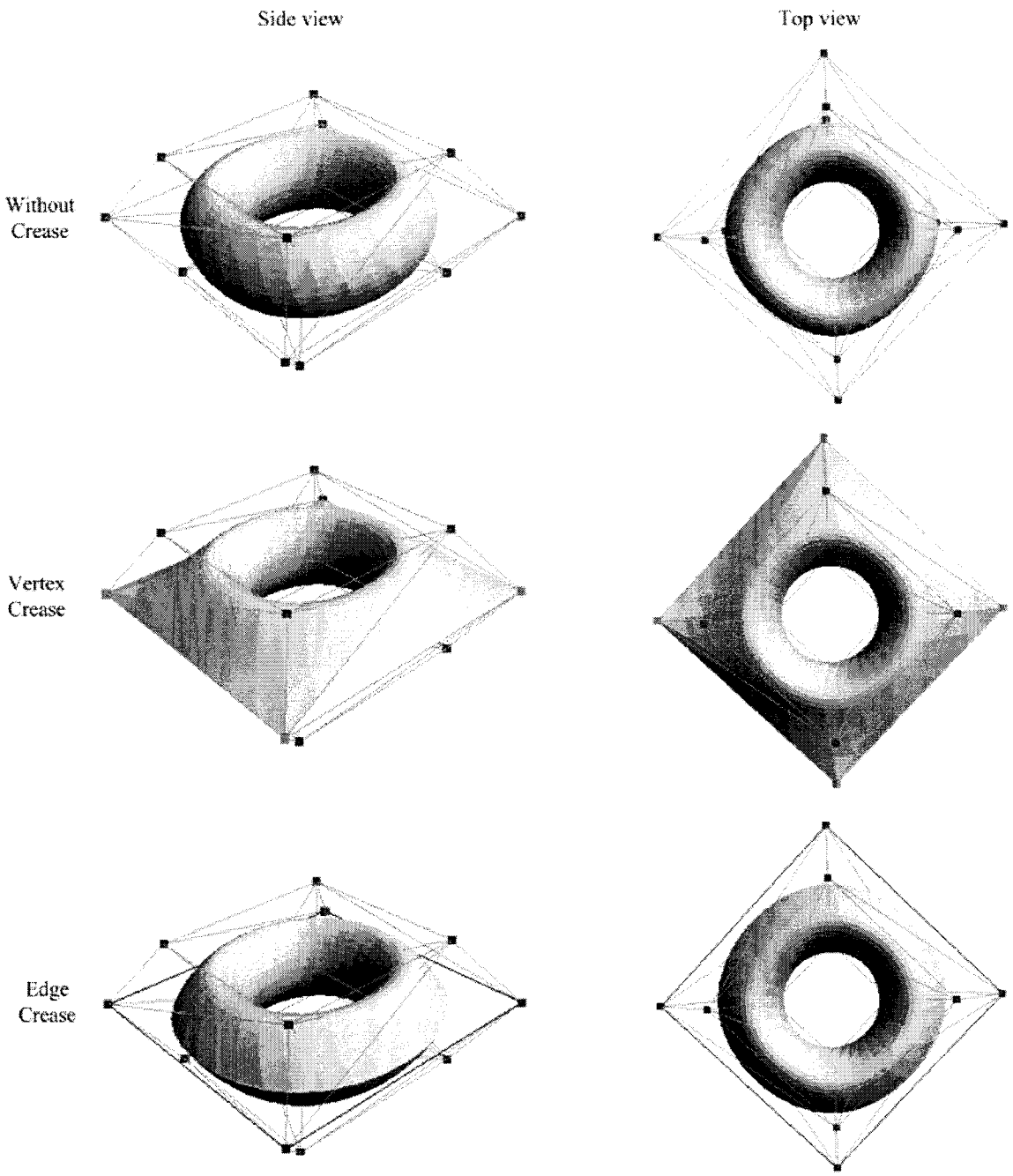


Figure IV -4: The illustration of defining vertex creases/edge creases in the middle of the Loop subdivision torus surfaces, where the surfaces inside the control meshes are subdivided 3 times from two different points of views.